

VISION Umwelt / UIS-Kern: Dynamische Integration hybrider Datenstrukturen¹

Thomas Bandholtz
Sema Group GmbH
Köln

Schlüsselworte

Umweltinformationssysteme, Führungsinformationssysteme, Datenbankentwurf, Objektorientierung, Integration, Dynamik des Informationsbedarfs.

Inhalt

1	Zwei Projekte - eine Technologie	2
2	Aufgabenstellung	3
2.1	Hybride Daten	3
2.2	Dynamik des Informationsbedarfs	4
2.3	Ergonomie	4
2.4	Portierbarkeit und Standards	5
3	Lösungsansatz	5
3.1	Ein "Container-Terminal" für Umwelt-Daten	5
3.2	Objektartenverzeichnis und Objektdaten	6
3.2.1	Ein Beispiel: Die Höhenlage von Biotopen	7
3.2.2	Weitere Eigenschafts-Klassen	9
3.2.3	Die Eigenschaftsklasse "Zeit"	10
3.2.4	Komplexe Eigenschaftsklassen	11
4	Visualisierung	13
5	Schlußbetrachtung	14
Literatur		15

Zusammenfassung

Ein objektorientierter Ansatz auf Basis des relationalen Modells ermöglicht die zeitnahe Integration heterogener Umweltdaten in ein ergonomisch und (geo-)graphisch anspruchsvolles Informationssystem. Die ständige Aktualisierung und Erweiterung von Datenbeständen ist dabei Teil der Anwendung des Systems und erfolgt ohne Eingriffe durch die Systementwicklung. Bestehende Anwendungen können sofort mit den neuen Daten arbeiten, ohne daß erweiterte Datenmodelle oder Bildschirmmasken erstellt werden müssen. Das System wird derzeit von den Umweltministerien zweier Bundesländer eingesetzt und arbeitet auf unterschiedlichen RDBMS (Oracle und Informix) unter UNIX und X11/Motif.

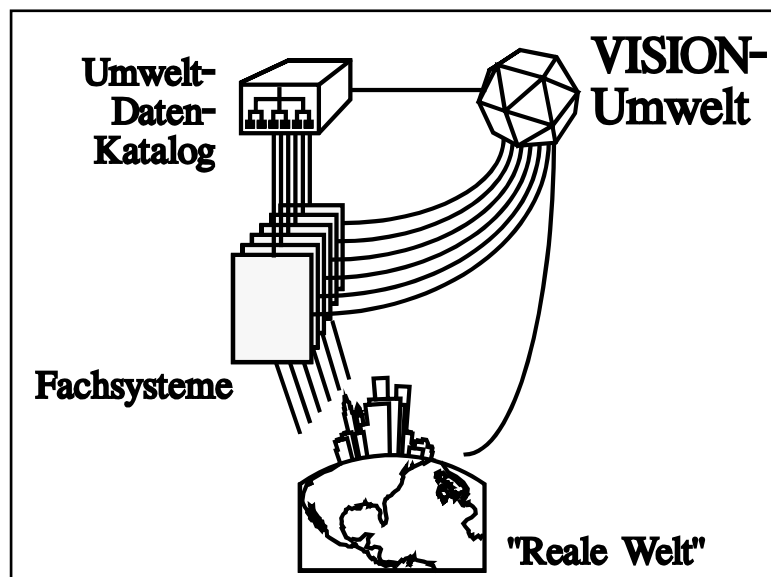
¹ Jahrestagung Deutsche ORACLE- Anwendergruppe e.V.. 1993

1 Zwei Projekte - eine Technologie

Im Folgenden wird von einer Datenbank-Anwendung berichtet, die zunächst für ein einzelnes Projekt individuell entwickelt wurde. Der Lösungsansatz hat sich hier bewährt und erschien schließlich so allgemeingültig, daß er inzwischen in einem weiteren Projekt eingesetzt wird. Es besteht darüber hinaus berechnete Aussicht, daß weitere Anwendungspartner hinzukommen werden.

Vision-Umwelt Niedersachsen

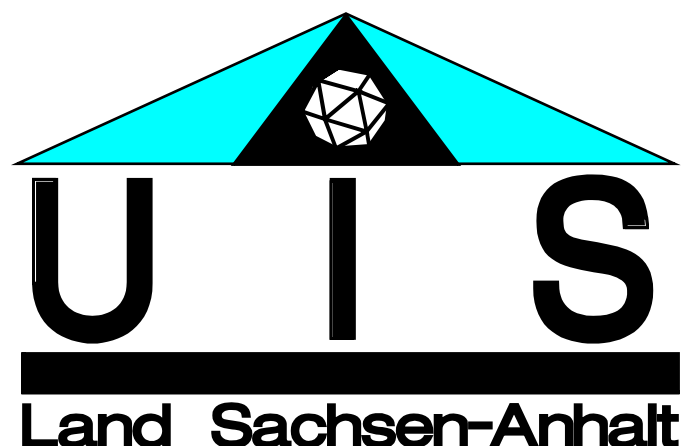
VISION-Umwelt wird seit 1990 als das Führungsinformationssystem innerhalb des Umwelt-Informationssystems der Umweltministeriums im Land Niedersachsen (NUMIS) entwickelt. 1992 entstand ein erster funktionsfähiger Prototyp, der in diesem Jahr zum produktionsreifen Pilotsystem ausgebaut wird. [1][2][3].



VISION-Umwelt führt ausgewählte Daten aus den Umwelt-Fachsystemen zusammen, wobei der Umweltdatenkatalog (UDK, ein weiterer NUMIS-Baustein) zusätzlich eine vollständige Übersicht über alle Fachdaten ermöglicht. Daneben "leistet" sich VISION den "Luxus" einer eigenen Sicht auf die reale Welt, auch an den Einzelsichten der Fachsysteme vorbei.

UIS-Kern Sachsen-Anhalt

Im Sommer 1993 fanden die Umweltministerien der Länder Niedersachsen, Sachsen-Anhalt und Hessen zu einer Kooperation zusammen, die eine gemeinsame Nutzung und Weiterentwicklung der mit VISION-Umwelt entwickelten Informationstechnologie beinhaltet.



Hat VISION in Niedersachsen die Rolle eines Dachsystems, so soll UIS-Kern in Sachsen-Anhalt den Kern des gesamten Umwelt-Informationssystems, also einschließlich der Fachsysteme selbst bilden. Im gegenwärtigen Projekt liegt der Schwerpunkt dabei in Anwendungen auf der Ebene des Führungsinformationssystems, die Datenhaltung erstreckt sich jedoch in zunächst "schmalen Säulen" bis tief in die Fachsysteme hinein. [4]

Hessen ist bisher Mitglied der Länderkooperation, noch ohne eigenes Anwendungsprojekt.

2 Aufgabenstellung

Trotz der Unterschiede zwischen den Einsatzgebieten liegt die Herausforderung in beiden Projekten grundsätzlich ähnlich:

- 1 hybride Daten,
- 1 hohe Dynamik des Informationsbedarfs,
- 1 hohe Ansprüche an die Ergonomie und
- 1 problemlose Portierbarkeit zwischen Systemplattformen auf der Basis realisierter Standards.

2.1 Hybride Daten

Dieser Begriff ist im Bereich der geographischen Informationssysteme schon fast zum Modewort geworden. Es bezeichnet dort speziell eine integrierte Bearbeitung geographischer Raster- und Vektordaten. Auch VISION-Umwelt/UIS-Kern ist in diesem Sinne als hybrides geographisches Informationssystem anzusehen.

Hybride bedeutet jedoch mehr als den Unterschied zwischen Raster- und Vektordaten. Ein Blick ins Lexikon belehrte mich seinerzeit, daß es sich schlicht um das griechische Wort für "Bastard" handelt. Und diese Deutung kam meinen Erfahrungen mit konkreten Umweltdaten schon sehr nahe.

Es geht hier kaum allein um die eine oder die andere Form definierter *geographischer* Daten als tatsächlich um ein kaum abzugrenzendes Rudel von Kreuzungen der verschiedensten Modellansätze für Daten jeder Art und jeden Inhalts, wobei die säuberliche Scheidung zwischen geographischen und nicht-geographischen Daten, die ja mancherorts so hervorgehoben wird (um beide dann "verknüpfen" zu können), nicht weiterhilft. Denn die Vergleichbarkeit und die Herstellbarkeit relationaler Bezüge ist auf beiden Seiten nicht gegeben, und dies gestaltet sich bei den nicht-geographischen Daten vielleicht sogar noch schwieriger. Geographische Eigenschaften von Objekten sind grundsätzlich nicht anderes zu behandeln als nicht-geographische.

Wir stehen vor dem Problem als Informatiker, hervorragendes lokales Expertenwissen eines Fachgebiets mit hervorragendem lokalem Expertenwissen eines anderen Fachgebiets in Beziehung setzen zu wollen - ohne selbst auf diesen Anwendungsgebieten wenigstens ein kleiner Experte zu sein.

Professionalität auf fachlichem Feld und auf dem Feld der Informatik gehen ja bis heute noch selten Hand in Hand. Und so finden wir auch in den "gewachsenen" Datensammlungen der einzelnen Fachthemen begreiflicherweise ein ebenso bescheidenes Maß an "sauberer" Modellierung wie, anders herum, bei den Modellen der Informatiker die fachliche Spezialität häufig dem vor formalen Modell zurückstecken muß.

Wir müssen in unseren Projekten nun versuchen, all die unterschiedlichen Bezugssysteme, der Verschiedenheit in der zeitlichen und räumlichen Auflösung, wie auch den Heerscharen von jeweils privat eingerichteten DBase-Tabellen gerecht zu werden.

Daneben gibt es einige entwickeltere Fachsysteme, die DV-technisch professionell organisiert sind. Von diesen kann man vieles lernen, und auch diese untereinander voneinander. Es ist dabei ganz folgerichtig, daß, wenn ich ein Fachsystem für den Sonderabfalltransport zu entwickeln habe, ich mich zunächst nicht darum kümmern kann, ob

etwa die Erfassung bedrohter Tier- und Pflanzenarten in vergleichbarer Weise vor sich gehen kann.

Solche Fachsysteme stellen der Integration fast noch härtere Anforderungen als der Wildwuchs der semi-spontanen PC-Tabellen: Sie sind vergleichsweise stabil und decken in einem selbst schon weitgesteckten Teilgebiet das gegebene Umfeld ab: es handelt sich hier um Systeme, mit denen ein integrativer Ansatz zu rechnen hat.

2.2 Dynamik des Informationsbedarfs

Hinzu kommt ein ungeheuer schneller Wandel der Relevanz, der den einzelnen Themen innerhalb der Umweltinformatik insgesamt zugewiesen wird. Auch dringt die Datenerhebung selbst in immer neue Gebiete vor.

Dagegen sieht das klassische Phasenmodell in der Softwareentwicklung ein nacheinander von Informationsanalyse, Modellierung und Realisierung vor.

Wenn solche klassischen Integrationsansätze scheitern, dann tun sie dies in erster Linie, weil sie eine frühe Momentaufnahme des Informationsbedarfs zugrunde legen und sich damit zum Zeitpunkt ihrer Fertigstellung auf Informationsstrukturen beziehen, die schon gar nicht mehr aktuell sind. Die stets begehrten "neuesten" Daten und Zusammenhänge lassen sich mit dem eigentlich ebenso neuen System schon nicht mehr darstellen und auswerten, und damit ist das System bereits veraltet, bevor es richtig in Betrieb gegangen ist.

Die notwendige Toleranz gegenüber einem sich verändernden Informationsbedarf und -angebot stellt dem Datenbankentwurf darüberhinaus eine scheinbar groteske Aufgabe: Er muß bereit sein, auch solche Daten und Sichten zu integrieren, die ihm noch gar nicht bekannt sind.

Der Zwang zur Offenheit gegenüber neuen Daten verhindert es, die aktuell gesehenen Zusammenhänge in Datenbanken einzufrieren, die zwar "nach Maß" relational modelliert, aber schon heute abend fachlich anachronistisch sind.

Erst unter diesem Druck entsteht die nötige Offenheit gegenüber den fragmentarischen Charakter der heute vorliegenden Informationsbestände.

2.3 Ergonomie

All diese Daten müssen natürlich jederzeit in der jeweils benötigten Auflösung vorliegen, und dies selbstverständlich unter einer einheitlichen, intuitiv zu bedienenden und visuellen Oberfläche. Man verzeihe mir den fast ironischen Ton. Diese Ansprüche sind ja begreiflich und berechtigt, im Sinne eines Umwelt-Informations-Managements existenziell notwendig.

Trotzdem wird jeder Kollege, der sich an einem solchen Thema versucht hat, mir wahrscheinlich beipflichten, daß er Momente kennt, in denen er glaubte, an der Quadratur des Kreises zu arbeiten.

Bei aller fachlichen Vielfalt und dem schnellen Wandel des Interesses benötigen wir ein schlankes, geradliniges System, daß mit wenigen Dialogen und Visualisierungstypen eine sichere Navigation und stets vertraute Darstellung der Abfrageergebnisse garantiert. Es geht nicht an, daß die Zahl der Selektions- und Ausgabemasken innerhalb weniger Jahre in die hunderte, wenn nicht tausende steigt, wie dies heute noch weitgehend üblich ist.

2.4 Portierbarkeit und Standards

Last but not least: Die Investitionssicherheit und in zunehmenden Maße auch die enge Kooperation zwischen Anwendern verlangt übertragbare, überlebensstarke Systeme. Der Lebenszyklus eines Führungsinformationssystems wird z.B. als langfristiger erkannt als der eines (beliebigen) Produktes der Bürokommunikation. Dies verbietet eine strategisches Bindung an ein solches Produkt.

Aber auch das Setzen auf die "amtlichen" Standards ist keine seligmachende Lösung. Am Beispiel von ANSI SQL Level 2, mit dem sich Oracle 7 ja - unwidersprochen - vollständig

konform erklärt, kann das leicht gezeigt werden. Der berechtigte Stolz auf die prompte Implementierung durch das gewählten RDBMS-Produkt verkehrt sich aus unserer Sicht paradoxerweise in die Bangigkeit einer faktisch fast proprietär wirkenden Lösung: Wenn ich mit Partnern arbeite, die andere RDBMS einsetzen, und diese unterstützen Level 2 noch nicht in gleichem Maße, droht mir die Isolation der einsamen Spitze - während die anderen, wenn auch weit abgeschlagen, behaglich kommunizieren können.

Diese Situation veranlaßt uns zu einem Anwendungs-Standard, der nur wenig oberhalb Level 1 liegt, also da, wo sich die meisten relevanten Datenbankhersteller normativ noch einigermaßen "unter einen Hut bringen lassen".

Daher muß ich Sie enttäuschen, wenn Sie hier einen avandgardistischen Bericht über den zukunftsweisenden Einsatz von Oracle 7 erwarten - wie arbeiten noch mit dem Leistungsumfang von Version 6, auch wenn in Niedersachsen die Version 7 derzeit (August 93) installiert wird.

Der Vollständigkeit halber sei kurz erwähnt, daß sich beide Projekte über ANSI Level "1¼" hinaus allein auf UNIX, X11 und OSF/Motif abstützen. Weitere Voraussetzungen an die Systemplattform bestehen nicht. Die Unterstützung von MS-Windows Clients durch UNIX-Server wird derzeit erprobt.

3 Lösungsansatz

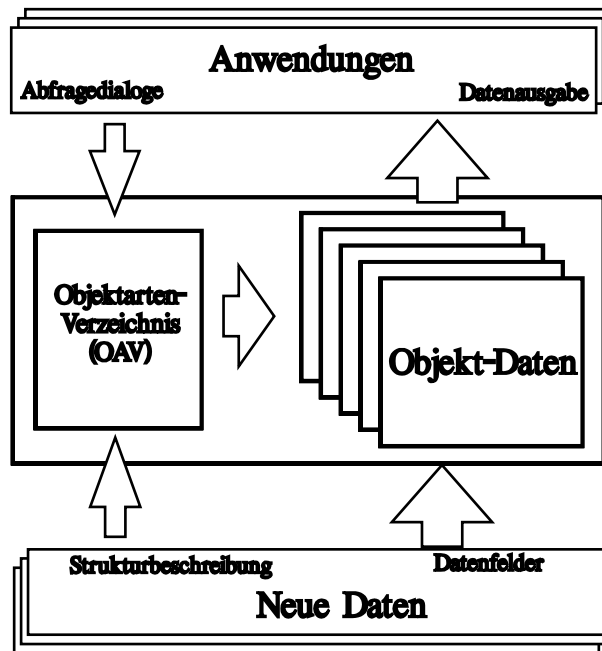
Kommen wir nun zum Lösungsansatz, und dies - wie es dem Tagungsanlaß entspricht - schwerpunktmäßig hinsichtlich des Datenbankentwurfs.

Zunächst haben wir, basierend auf der Informationsanalyse, ein *konzeptuelles Schema* mittels Entity-Relationship Methodik entworfen. Ich will hier nicht im einzelnen darauf eingehen, da es die Kernprobleme der Hybridität und Dynamik noch nicht bewältigen konnte. Interessierte können gern ein Exemplar des Feinkonzepts von 1991 anfordern [1].

Der Lösungsansatz konzentriert sich auf den Übergang in das *interne Schema*, dessen Grundzüge ich Ihnen an einigen Beispielen erläutern möchte.

3.1 Ein "Container-Terminal" für Umwelt-Daten

Ausgangspunkt war die Idee, nicht die Modellierung eines definierten Informationsbestands anzubieten, sondern eine Art "implementierte Methode", mit der sich die jeweiligen Ausprägungen der aktuellen Informationsbestände *ohne Änderungen des Datenmodells* komfortabel und sicher erfassen lassen. Herausgekommen ist etwas, das ich plakativ gerne als "Container-Terminal für Umweltdaten" beschreibe.



Die Abbildung zeigt eine grobe Struktur aus Anwendungen, integrierten Daten und Quelldaten. Der eigentliche "Container-Bahnhof" liegt in der Mitte: Hier sehen wir zum einen ein "Objektarten-Verzeichnis" (OAV), welches als eine bestimmte Ausprägung der Idee des "Data Dictionary" verstanden werden kann, und zum anderen einen *dynamischen* Satz konkreter Objektdaten, deren aktuelle Struktur *eindeutig* aus dem aktuellen, deskriptiven Inhalt des OAV ableitbar ist.

Die Anwendungen greifen zunächst ausschließlich auf das OAV zu, also auf die aktuelle Datenbeschreibung, und parametrisieren sich dabei so, daß sie auf die momentanen konkreten Daten vollständig eingehen können.

Auf der anderen Seite stehen die hybriden Quell- oder Rohdaten, die es zu integrieren gilt. Auch hier wird zunächst ausschließlich eine Strukturbeschreibung im OAV abgelegt, woraufhin eine zwingende Tabellenstruktur für die Aufnahme der Daten selbst gefolgt werden kann.

Damit das System funktioniert, müssen die Daten nach gewissen normativen Regeln beschrieben werden, sie müssen gewissermaßen in "genormte Container" verpackt werden, mit denen die Anwendungen umgehen können.

Der Schlüssel für eine anwendungs- und datengerechte normative Struktur liegt in einem Satz abstrakter Datentypen, die mit Methoden verknüpft sind. Wir nennen sie hier *Klassen von Objekteigenschaften*. Jede dieser Klassen ist gewissermaßen ein Sub-Container-Modell, wobei ein einzelner Datenbestand dann beschrieben werden kann als eine definierte Menge "beschrifteter Sub-Container", von denen jeder einem der bekannten Modelle folgt. Mit anderen Worten: es handelt sich um eine definierte Menge parametrisierter Instanzen der Klassen von Objekteigenschaften. Dem Kenner von Motif mag hier eine gewisse Ähnlichkeit mit dem Ressourcen-Konzept der Widget-Klassen auffallen.

3.2 Objektartenverzeichnis und Objektdaten

Das OAV wie auch die Objektdaten bestehen aus eindeutig definierten relationalen Tabellen, wobei die Tabellen des OAV in Anzahl und Feldstruktur konstant sind, die der Objektdaten jedoch dynamisch. Ich möchte dies anhand einfacher Beispiele nachvollziehbar machen. Hierfür zunächst ein Überblick über einige wichtige Tabellen des OAV:

- OAV-OA Die Liste der aktuellen Einzeldatenbestände, hier "Objektarten" genannt.
- OAV-ATT Die Liste aller Eigenschaften der aktuellen Objektarten, kodiert nach der zugrundeliegenden Klassenbildung

- OAV-ATT-ZAHL Parametrisierung aller Objekteigenschaften der Klasse "Zahl"
- OAV-ATT-ZEIT Parametrisierung aller Objekteigenschaften der Klasse "Zeit"
- OAV-ATT-R1 Parametrisierung aller Objekteigenschaften der Klasse "Einfach-Relation"
- OAV-ATT-RN Parametrisierung aller Objekteigenschaften der Klasse "Mehrfachrelation"

Sicher gibt es viele weitere Tabellen im OAV, so z.B. für die Parametrisierung weiterer Eigenschafts-Klassen, aber auch für die Anwendungssteuerung (Freigabe bestimmter Objektarten bzw. ihrer Eigenschaften für bestimmte Anwendungen), zur Benutzerverwaltung (logische Zugriffsrechte) und zur Verwaltung und operativen Steuerung der Datenaktualisierung über Importfilter.

Die wenigen hier aufgeführten Tabellen reichen jedoch aus, um die Prinzipien des Vorgehens zu erkennen zu machen.

3.2.1 Ein Beispiel: Die Höhenlage von Biotopen

Im Folgenden soll ein einfaches Beispiel erläutern, wie eine Objektart mit einer Eigenschaft definiert wird und was das System mit dieser Definition anfängt.

OAV-OA (Liste der Objektarten)

OA-ID	Name	AttrTab	
biot	"Biotope"	oa-biot	...

Im obigen Tabellenbeispiel ist vereinbart, daß es eine Objektart namens "Biotope" gibt. Sie hat den internen Schlüssel "biot", und der Zugang zu den Objekten selbst erfolgt über eine Attribut-Tabelle oa-biot.

Diese Informationen sind allein in der Tabelle OAV-OA enthalten. Greift das System darauf zu, so erwartet es, daß es die Tabelle oa-biot gibt. Durch die bloße Benennung in OAV-OA wissen wir noch wenig über ihre Struktur. Bekannt sind bisher lediglich ihr Name und - durch eine implizite Vereinbarung, die für alle Attribut-Tabellen gilt - zwei ihrer Felder, nämlich die ID und der Name der Objekte selbst, hier also der Biotope.

oa-biot (Liste der Biotope mit ihren einfachen Eigenschaften)

O-ID	Name	
4711	"Waldau"	...

Wir können aber mit einer bloßen Liste von Biotopen wenig anfangen, wenn wir keine Informationen über weitere Eigenschaften haben. Wir vereinbaren also zunächst eine solche Eigenschaft, nämlich im Beispiel die "Höhe". Dies geschieht in der Tabelle OAV-ATT. Dort wird auf den Schlüssel der zugehörigen Objektart verwiesen (biot), und diese Eigenschaft erhält einen internen Schlüssel, der innerhalb einer Objektart eindeutig zu sein hat. Der Einfachheit halber verwenden wir hier eine Ableitung des Eigenschaftsnamens, aus "Höhe" wird die ID "hoehe". Schließlich finden wir ein viertes Feld, den Attribut-Typ, welcher hier mit "zahl" angegeben ist.

OAV-ATT (Liste aller Objekt-Attribute)

OA-ID	ATT-ID	Name	AttrTyp
biot	hoehe	"Höhe"	zahl

Beim Attribut-Typ "zahl" handelt es sich offensichtlich noch nicht um einen Datentyp auf RDBMS-Ebene, sondern um einen logischen Typ aus Sicht der Anwendung. Wir nannten diese Typen auch "Klassen von Eigenschaften".

Jeder dieser Klassen verfügt über eine eigene Tabelle, welche die allgemeine OAV-ATT spezifisch ergänzt und letztlich den Zugriff auf die Werte der betreffenden Attribute

ermöglicht. In diesem Fall handelt es sich um die Tabelle OAV-ATT-ZAHL. Hier wird zunächst auf die Eigenschaft verwiesen (OA-ID und ATT-ID als zusammengesetzter Schlüssel). Das Feld *Dezimal* enthält die Anzahl der Nachkommastellen - hier 0, da es sich um eine Ganzzahl handeln soll. Das Feld *Einheit* gibt die Maßeinheit wieder. Das Feld mit dem Namen *Feld* schließlich bezeichnet den Namen einer Tabellenspalte ("hoehe"), in der die Werte abgelegt sind.

OAV-ATT-ZAHL (Beschreibung der Zahlen-Attribute)

OA-ID	ATT-ID	Dezimal	Einheit	Min	Max	Feld
biot	hoehe	0	"m"	0	400	hoehe

Da es sich hier um ein einfaches Zahlen-Attribut handelt, können wir diese Spalte zweckmäßig in der Attribut-Tabelle oa-biot unterbringen. Diese sieht nun wie folgt aus:

oa-biot (Liste der Biotope mit ihren einfachen Eigenschaften)

O-ID	Name	hoehe	
1	"Waldau"	220	...

Mit den Angaben in OAV-ATT-ZAHL kann aber auch ein Dialog versorgt werden, mit dem der Anwender Biotope über ihre Eigenschaft "Höhe" selektieren kann. Dieser sieht dann so aus:



Hier gehen auch die bisher nicht erläuterten Felder *Min* und *Max* aus OAV-ATT-ZAHL ein. Es handelt sich um die Grenzen eines für diese Eigenschaft gültigen Wertebereichs, wobei hier einmal dahingestellt sein soll, ob um einen tatsächlichen oder einen normativen (beides ist möglich).

Bei diesem Dialog handelt es sich um einen *allgemeinen* Dialog für Eigenschaften vom Typ Zahl - ganz gleich ob es sich um die Höhenlage von Biotopen handelt oder um irgendeine andere Zahl im System. Es bleibt stets derselbe Dialog, welcher mit den spezifischen Angaben aus dem Objektartenverzeichnis vorbelegt und kontrolliert wird (Bezeichnung, Einheit, Nachkommastellen und Wertebereich) .

Ebenso formal und gleichgültig gegenüber dem eigentlichen Inhalt der Eigenschaft, kann die Entscheidung des Anwenders in SQL-Klauseln umgesetzt werden. In der Abbildung fragt der Anwender nach den Biotopen, die mehr als 200 m über Normalnull liegen. Nach allem, was den bisherigen Tabellenbeispielen entnommen werden kann, "weiß" das System nun, daß es formulieren muß:

where oa-biot.hoehe > 200

Das System weiß auch, daß die Tabelle oa-biot in die FROM-Klausel des ganzen Statements aufgenommen werden muß, und es weiß natürlich auch, daß nur die erste WHERE-Bedingung mit where beginnt und weitere mit and oder or anzuhängen und gegebenenfalls logisch zu klammern sind.

3.2.2 Weitere Eigenschafts-Klassen

Dieses Beispiel ist prototypisch für alle weitere *einfachen* Attribut-Typen bzw. Eigenschaftsklassen. Als "einfach" bezeichnen wir sie deshalb, weil sie durch eine einfache Spalte in der Attribut-Tabelle unterzubringen sind. Weitere einfache Eigenschaftsklassen sind neben Zahl z.B.

1 Wahrheit

"ja" oder "nein" - der sogenannte Boolesche Wert

1 Text

eine Zeichenkette variabler Länge

1 Zeit

ein Zeitpunkt von grundsätzlich beliebiger Genauigkeit

1 1:1 Relation

Relationale Beziehung eines Objekts zu genau einem Objekt einer anderen Objektart.

Daneben gibt es, wie man sicher erwartet, auch *komplexe* Eigenschaftsklassen. Diese können nicht mehr in einer Spalte der Attributtabelle abgelegt werden, sondern benötigen separate Tabellen, auf die über Joins zugegriffen werden muß. Solche Eigenschaften sind z.B.

1 1:n Relation

Relationale Beziehung eines Objekts zu mehreren Objekten einer anderen Objektart.

1 Relation mit Eigenschaften

Die Relation hat selbst weitere Eigenschaften der bekannten Klassen.

1 Wertreihe

Meß- oder statistische Reihe

1 Topographie

Kartographische Geometrien

1 Virtuelle Eigenschaft

Schnittstelle zum Einbinden von Rechenfunktionen, welche den Wert der Eigenschaft aktuell errechnen.

Es würde den Rahmen dieses Vortrags sprengen, wenn wir hier auf all diese Klassen im einzelnen eingehen würden. Lassen Sie mich eine weitere Klasse herausgreifen, die in einigen Aspekten typisch für den verwendeten Ansatz ist: die Klasse "Zeit". Hierbei werde ich auch genauer auf den dahinter liegenden RDBMS-Datentyp eingehen.

3.2.3 Die Eigenschaftsklasse "Zeit"

Der Datentyp DATE von Oracle (wie auch die entsprechenden Lösungen anderer Hersteller) hat aus der Sicht unserer Anwendung eine entscheidende Schwäche: Ich muß die Angaben stets taggenau spezifizieren: Eine Eingabe wie z.B.

```
insert into .... values (to_date('10.1992', 'mm.yyyy'))
```

wird zwar akzeptiert; die Abfrage auf dasselbe Feld zeigt jedoch, daß Oracle stillschweigend die beabsichtigte Aussage

"im Oktober 1992"

durch

"am 1. Oktober 1993"

ersetzt hat.

Dieses Verhalten ist für die Angabe der zeitlichen Gültigkeit von Umweltinformationen nicht akzeptabel. Nun könnte man durch ein zusätzliches Feld zu erkennen geben, wie genau das Datum beachtet werden soll, oder aber das Datum ganz in einzelne Felder für Tag, Monat, Jahr auflösen und dort mit NULL-values arbeiten. Wir haben einen dritten Weg gewählt: wir verwenden einen Zeit-"String" variabler Länge in der Form:

```
"yyyy[.mm[.dd[.hh[.mm[...]]]]]"
```

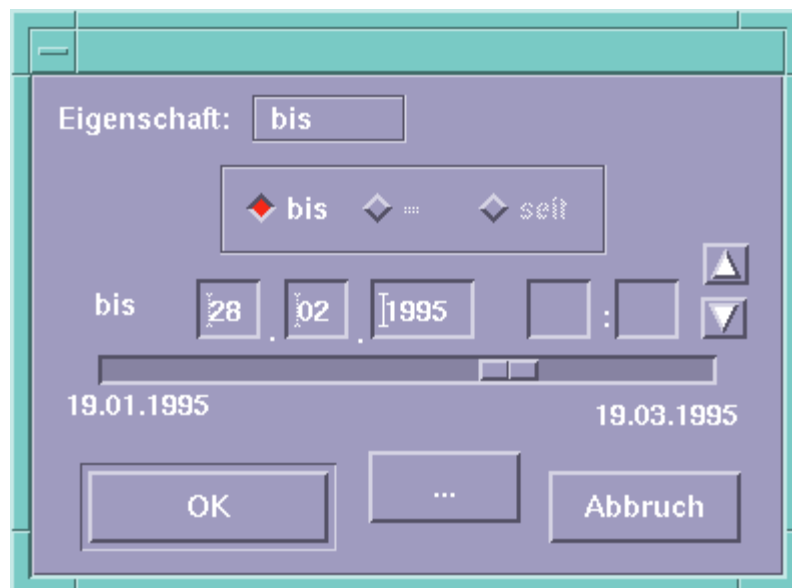
"1992" heißt also "im Jahr 1992", "1992.10" heißt "im Oktober 1992", usw. Prinzipiell ist die Genauigkeit unbegrenzt. Aus Sicht unserer Anwendung werden kaum genauere als Minutenwerte gebraucht werden. Wir legen dieses Feld daher in einem VARCHAR von maximal 255 Zeichen Länge ab, so daß reichlich Spielraum für Genauigkeit gegeben ist.

Mit dieser Lösung sind wir hinreichend variabel, um unterschiedliche Genauigkeiten sogar in ein und derselben Spalte mit denselben Methoden zu behandeln.

Abfragen erlauben über den Einsatz der LIKE-Klausel eine große Flexibilität wie z.B. LIKE '1992%' (im Jahr 1992 - egal wie genau angegeben), LIKE '1992.%' (im Jahr 1992, aber genauer angegeben), LIKE '????'.10%' (alle Oktoberwerte in allen Jahren) usw.

Innerhalb von Spalten mit gleicher Genauigkeit funktionieren auch die Vergleichsoperatoren < und > in natürlicher Weise. "1992.10.01" ist größer (und später) als "1992.09.31" - und zwar im Vergleich von Zeichenketten, wie auch hinsichtlich des Zeitpunktes (später). Probleme könnten sich allein beim Vergleich zwischen Daten verschiedener Genauigkeit ergeben: "1992.10" gilt im Zeichenkettenvergleich als größer als "1992", der Oktober 1992 kann aber in unserem Sinne nicht als später als das Jahr 1992 angesehen werden. Dieses Problem existiert aber bei jeder Form eines Datentyps für Zeit (auch bei DATE) und ist allein durch die Anwendung zu lösen.

Der Zeit-Dialog gibt hier die jeweils gewünschte Genauigkeit vor, indem genau die betreffenden Teilfelder der Gruppe Jahr-Monat-Tag-Stunde-Minute sensitiv sind. Ansonsten ist er dem Zahlendialog sehr ähnlich.



Auch die Tabellenstruktur ist ähnlich wie die der Zahl: die entsprechenden Definitionen stehen in der Tabelle OAV-ATT-ZEIT, und die Zeit-Daten selbst finden wir in einer VARCHAR-Spalte der Attributtabelle.

3.2.4 Komplexe Eigenschaftsklassen

Am Übergang von den einfachen zu den komplexen Eigenschaften liegen die einfachen Relationen nach dem Muster:

Jedes Biotop liegt in einem Landkreis.

Da es sich ja nur um einen Landkreis je Biotop handelt, kann diese Eigenschaft ohne weiteres in einer weiteren Spalte der Attribut-Tabelle untergebracht werden, neben der Höhe und dem Zeitpunkt der Erfassung. Diese sähe dann z.B. bisher so aus.

oa-biot (Liste der Biotope mit ihren einfachen Eigenschaften)

O-ID	Name	hoehe	zeitpunkt	landkreis	
4711	"Waldau"	200	1992.10	4712	...

Nun steht aber unter *landkreis* nicht der Name eines Landkreises wie z.B. Celle oder Oldenburg, sondern eine Fremdschlüssel, der auf eine zweite Objektart neben den Biotopen hinweist. Landkreise sind ja nicht allein als Eigenschaft von Biotopen interessant, sondern sie sind selbst in der Tabelle OAV-OA registriert, verfügen selbst über einen Satz von Eigenschaften und auch über eine Attributtabelle.

Es kommt also -aus Sicht der Biotope- eine zusätzliche Tabelle ins Spiel, in welcher der Name des Landkreises zu finden ist. Dies ist eben die Attributtabelle der Landkreise. Zur Vermeidung von Redundanzen erhält die Eigenschaft "Landkreis" der Biotope daher einen Verweis auf den Objektarten-Schlüssel der Landkreise in der OAV-OA.

OAV-ATT (Liste aller Objekt-Attribute)

OA-ID	ATT-ID	Name	AttrTyp
biot	hoehe	"Höhe"	zahl
biot	zeitpunkt	"Zeit"	zeit
biot	landkreis	"Landkr."	r1

OAV-ATT verweist auf den Typ "r1" (= jedes Biotop zu 1 Landkreis). Dazu gehört die Tabelle OAV-ATT-R1, die wie folgt aufgebaut ist:

OAV-ATT-R1 (Beschreibung der R1-Attribute)

OA-ID	ATT-ID	R-OA-ID	Feld
biot	landkreis	lacr	landkreis

Beides zusammen bedeutet: Biotope haben eine Eigenschaft "Landkreis" als einfache Relation zur Objektart mit dem Schlüssel "lacr", und der Objektschlüssel des jeweiligen Landkreises findet sich in der Attributtabelle OA-BIOT im Feld "landkreis".

Komplizierter wird die Situation, wenn die Relation nicht in diesem Sinne einfach, sondern *mehrfach* ist, wenn z.B. jedes Biotop aus mehreren Vegetationstypen besteht, wie es die betreffende Kartieranleitung des Naturschutzes vorsieht [5]. Solche Vegetationstypen werden dort als "Erfassungseinheiten" bezeichnet.

Zunächst sind die Erfassungseinheiten im OAV als eigene Objektart organisiert, wie die Landkreise. Entsprechend erfolgt der Verweis auf deren Objektartschlüssel, um darüber deren Attributtabelle zu finden, welche die Namen enthält. Hier verfügt jedes Biotop aber nicht über genau einen, sondern über 0 bis 6 Verweise. Diese können im relationalen Modell nicht mehr als einfache Spalten in der Attributtabelle der Biotope abgelegt werden. Wir benötigen hierfür eine eigene "Relationentabelle", welche die Biotope mit einer variablen Anzahl von Erfassungseinheiten verknüpft.

Die Vereinbarungen im OAV sehen folgendermaßen aus:

OAV-ATT (Liste aller Objekt-Attribute)

OA-ID	ATT-ID	Name	AttrTyp
biot	hoehe	"Höhe"	zahl
biot	zeitpunkt	"Zeit"	zeit
biot	landkreis	"Landkr."	r1
biot	erfein	"Erfass..."	rn

"rn" ist der interne Code für eine Eigenschaft in der Form einer 1:n-Relation. Diese wird in der Tabelle OAV-ATT-RN weiter beschrieben:

OAV-ATT-RN (Beschreibung der Mehrfach-Relationen)

OA-ID	ATT-ID	R-OA-ID	Min	Max	RelTab
biot	erfein	erfein	0	6	biot-erf

Die Anwendung kann daraus lesen:

Die Eigenschaft "Erfassungseinheiten" der Biotope hat den Attribute-Schlüssel "erfein" und bezieht sich auf eine Objektart mit dem Objektarten-Schlüssel "erfein". Jedes Biotop zeigt auf mindestens 0 und höchstens 6 solcher Objekte, und diese Relationen sind in einer Tabelle mit dem Namen "biot-erf" gespeichert.

Implizit ist der Anwendung auch bekannt, wie solche Relationen-Tabellen strukturiert sind: sie bestehen stets aus zwei Feldern: eines für die Objekt-ID der attribuierten Objektart (O-ID, hier: Biotope), und ein weiteres für die Objekt-ID der referenzierten Objektart (R-O-ID, hier: Erfassungseinheiten)

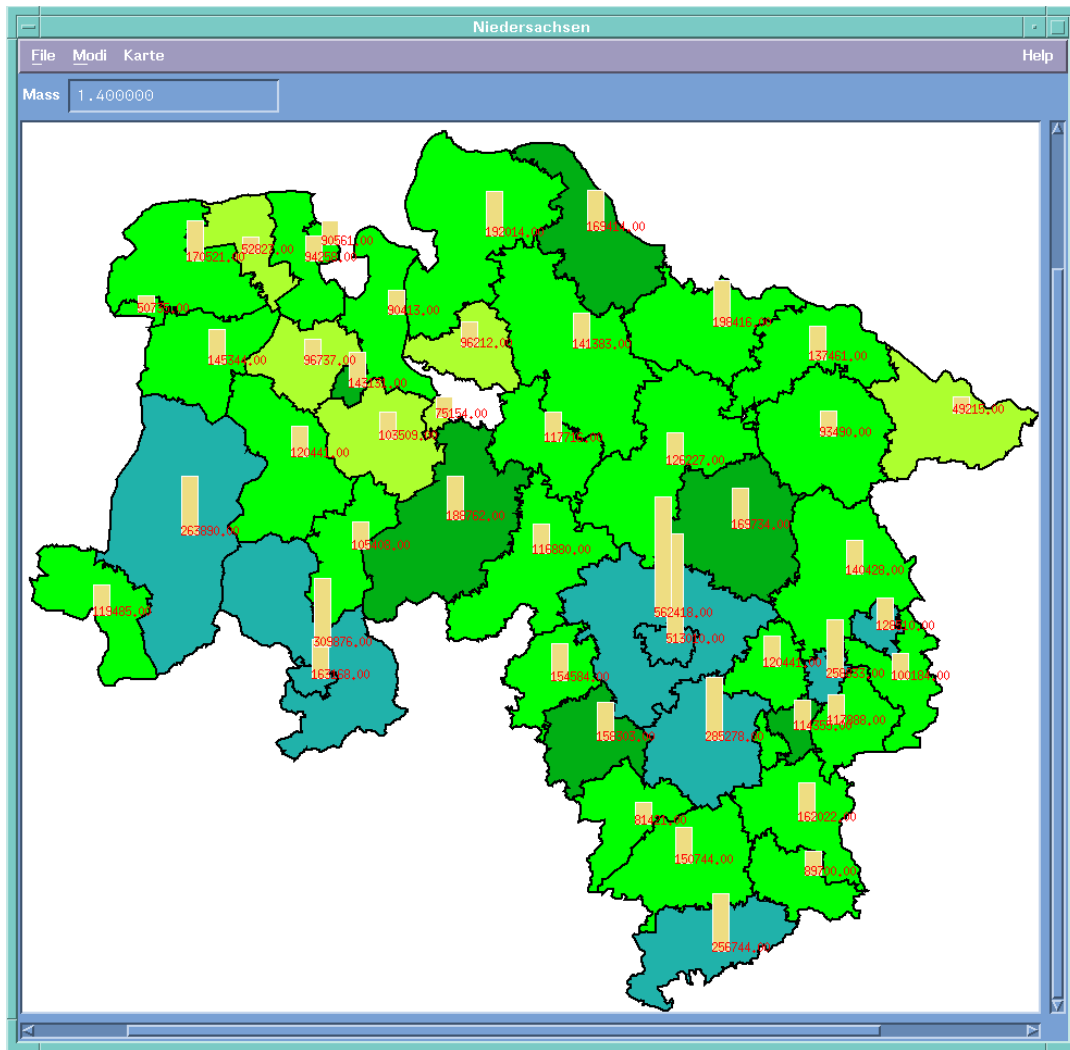
biot-erf (Liste aller Erfassungseinheiten der einzelnen Biotope)

O-ID	R-O-ID
4711	5812
4711	5813
4711	5819
4713	5813

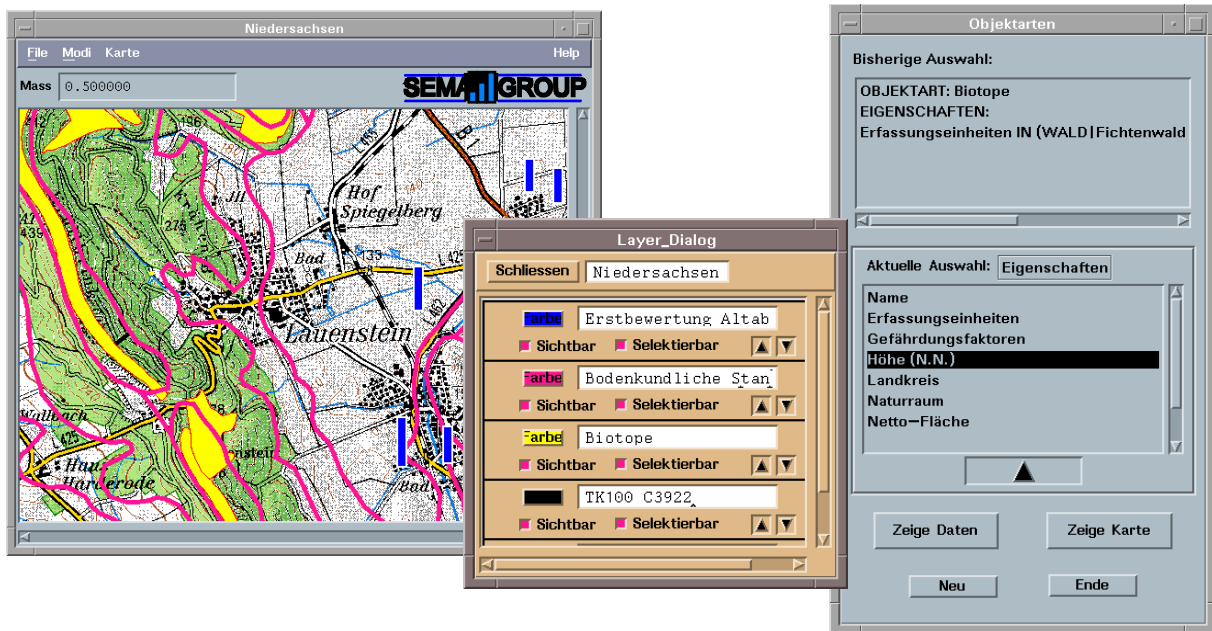
Dieses Beispiel beinhaltet 3 Erfassungseinheiten für das Biotop 4711 und eine für das Biotop 4713. Gibt es ein Biotop 4712, so ist diesem keine Erfassungseinheit zugewiesen.

4 Visualisierung

Die zur Verfügung stehende Zeit reicht nicht aus, um den Weg von diesen Datenstrukturen zu den Visualisierungen von Abfrageergebnissen nachzuvollziehen. Es sei hier nur darauf hingewiesen, daß mit einfachen Mitteln intuitiver Navigation komplexe Darstellungen der aktuellen Bestände erzielt werden können, von denen zweifellos die am interessantesten sind, welche vor kartographischem Hintergrund erfolgen. Hier zwei Beispiele: (im Schwarz-Weiß-Druck nicht angemessen ablesbar - ein solches System kommt ohne den Informationsträger Farbe nicht mehr aus).



Hier wird eine Farbstufen-Skalierung mit einer beschrifteten Balkenskalierung kombiniert, um zwei Themen aus der Regionalstatistik (..... und , bezogen auf Landkreise) in Beziehung zu setzen. Dasselbe Verfahren wird z.B. auch für die Visualisierung von Meßnetzen eingesetzt (flächenhafte Ausbreitung und punktuelle Meßergebnisse).



Hier werden verschiedene Themen vor dem Hintergrund einer topographischen Karte 1:100 000 zusammengeführt: Biotop unterschiedlicher Klassifizierung, bodenkundliche Standorte, Altablagerungen und Ergebnisse von Einzelmessungen. Die Damit aufgedeckten räumlichen Beziehungen lassen sich direkt aus der Darstellung für weitergehende Abfragen einsetzen.

5 Schlußbetrachtung

Die obige Darstellung des OAV ist noch immer sehr vereinfachend. Tatsache ist z.B., daß den Erfassungseinheiten der Biotop weiterhin ein jeweiliger Flächenanteil innerhalb des Biotop zugewiesen wird. Darüberhinaus haben wir uns nicht mit Meßreihen, Geometrien und virtuellen Eigenschaften beschäftigt. Diese Strukturen bauen auf den beschriebenen auf und führen sie konsequent weiter, sodaß schließlich jedes bekannte Datum eines Umwelt-Informationssystems abgebildet und formal bearbeitet werden kann.

Was hier gezeigt werden sollte, ist allein, daß eine Anwendung *ohne Rücksicht auf die konkreten Daten* programmiert werden kann, wenn diese sich konsequent formal beschreiben lassen.

Gegen diesen Ansatz ließe sich vor allem zweierlei ins Feld führen:

1. Die konkreten Daten, die von einem solchen System verarbeitet werden sollen, müssen zunächst in die von diesem System vorgegebenen Strukturen transformiert werden. Damit verlagert sich der Aufwand lediglich von der Anpassung der Anwendungen an die Daten in die Anpassung der Daten an die Anwendungen.
2. Es handelt sich beim Ansatz des OAV um einen Vorgriff auf zukünftige ANSI-konforme Regelungen für objektorientierte Erweiterungen des relationalen Modells und somit um eine proprietäre Lösung, die im Zuge der späteren Normungen anachronistisch werden kann.

Beide Einwände sind ernst zu nehmen und vor der Entscheidung für den vorgestellten Ansatz erwogen worden. Hier eine Reihe von Gegenargumenten:

- 1 Die Transformation von vielfältig strukturierten Datenbeständen in die Strukturen einer konstanten Anwendung ist grundsätzlich effektiver als die stetige Erweiterung der Anwendung durch Anpassungen an 1001 Strukturvariante der verzweigten Datenmodelle.

- 1 Die Transformation bestehender heterogener ("gewachsener") Datenstrukturen ist für eine integrierte Bearbeitung in jedem Fall erforderlich. Es geht hier also nicht darum, ob ein neues, integratives Datenmodell gebildet wird, sondern allein darum, wie.
- 1 Da nie von einem endgültigen Bestand an zu integrierenden Quelldaten auszugehen ist, muß ein solches Datenmodell von jedem historisch gültigen Bestand abstrahieren, auch wenn damit zunächst eine aktuelle Integration auf scheinbar unnötig abstrakten Niveau erfolgt.
- 1 Da auch weiterhin Quelldaten neu erfasst und modelliert werden, muß ein integratives Datenmodell geeignet sein, auch für diese Modellierung der Quelldaten selbst gelten zu können, so daß dann eine Transformation sich erübrigt.
- 1 Die integrierte Bearbeitung heterogener Quelldaten kann nicht - besonders im Umweltsektor - auf die Ergebnisse eines langfristigen Normierungsprozesses warten (z.B. ANSI Level 3), der danach schließlich auch erst noch von Produktanbietern konform implementiert werden will.
- 1 Der proprietäre Charakter des vorgestellten Modells bindet nicht an eine Systemplattform oder an einen Datenbankhersteller - proprietär ist hier lediglich die Anwendung selbst - wie jede Anwendung.

Diese Argumente können selbstverständlich nur als Denkanstöße gelten. Ein weiteres, gewichtigeres Argument finden Sie in der Vorführung von VISION-Umwelt / UIS-Kern in der begleitenden Ausstellung dieser Tagung.

6 Literatur

- [1] Feinkonzept 91
- [2] Fachliches Feinkonzept 92
- [3] DV-technisches Feinkonzept 92
- [4] Grobkonzept UIS-Kern 93
- [5] Kartieranleitung Biotope

Thomas Bandholtz
Sema Group GmbH
Siegburger Straße 215
50679 Köln

Tel. (0221) 8299-264
Fax. (0221) 8299-266