Thema

GIS-basierte, automatische Erfassung natürlicher Fliessgewässerhierarchien und ihre Abbildung in Datenbanken, beispielhaft dargestellt am Einzugsgebiet der Salza

Dissertation

zur Erlangung des akademischen Grades

Dr. rer. nat.

vorgelegt der

Mathematisch-Naturwissenschaftlich-Technischen Fakultät
(mathematisch-naturwissenschaftlicher Bereich)
der Martin-Luther-Universität Halle-Wittenberg

von Thomas Koschitzki

geb. am: 29.11.1966 in: Roßlau

Gutachter:

1. Prof. Dr. Manfred Frühauf, Martin-Luther-Universität Halle-Wittenberg
2. Prof. Dr. Bernd Diekkrüger, Universität Bonn
3. Prof. Dr. Rainer Duttmann, Christian-Albrechts-Universität zu Kiel

Die Verteidigung der Dissertation erfolgte am 16. Juni 2004.

urn:nbn:de:gbv:3-000007179
[http://nbn-resolving.de/urn/resolver.pl?urn=nbn%3Ade%3Agbv%3A3-000007179]

Inhaltsverzeichnis

Verzeichnis der Abbildungen

Unter-Oberlieger-Beziehungen

Implementierung klassischer Flussordnungsprinzipien

Der WEG-Schlüssel der Länderarbeitsgemeinschaft Wasser (LAWA)

Einführung in die Graphentheorie

Generischer Verzweigungscode als Grundlage der Hierarchisierung

Verschachtelte Hierarchien („Nested Sets“)

Hierarchisierung flächen- und punkthafter Elemente

Regionalisierung / Globalisierung

Nicht-baumstrukturierte Netzwerke (Routing-Algorithmen)

Anwendung zu einem hierarchiebasierten Fließgewässer- und Einzugsgebietsmanagement-System

Verzeichnis der Tabellen

Danksagung

An dieser Stelle möchte ich mich bei Prof. Dr. Manfred Frühauf für die freundliche Betreuung und vielfältige Unterstützung sowie den Freiraum während der Bearbeitungszeit bedanken.

Prof. Dr. Walter Gläßer und Dr. Thomas Wieser danke ich für die Ermöglichung der Tests mit SDE und Oracle am UFZ in Halle. Die fachliche und organisatorische Unterstützung durch Thomas hat mir besonders in der Anfangsphase der Arbeit sehr geholfen.

Den Mitarbeitern der Arbeitsgruppe, ganz besonders Daniel Wurbs und Reiko Liermann, danke ich für den häufigen Gedankenaustausch, mit dem so manche, auch unkonventionelle Problemlösung verbunden war, sowie Dr. Martin Sauerwein für die Auseinandersetzung mit der Arbeit und den beigesteuerten Ratschlägen. Insgesamt wurden in diesem Rahmen nicht nur fachliche Probleme gelöst, sondern auch Wert auf geselliges Beisammensein gelegt, was mir, zusammen mit der freundlichen, produktiven Arbeitsatmosphäre in sehr guter Erinnerung bleiben wird.

Steffen Kussmann und Ulf Nilius vom RP Halle (Abteilung 4, Dezernat 42 - Wasserwirtschaft) möchte ich für die Unterstützung bei der Datenbeschaffung, den fachlichen und oft auch persönlichen Austausch und das stete Interesse an den Fortschritten der Arbeit danken.

Den Mitarbeitern des TULG, des PIK sowie des LUA Brandenburg bin ich ebenfalls für die freundliche Bereitstellung der Datengrundlagen dankbar.

Dr. Bernd Pfützner vom BAH danke ich für die persönlichen und fachlichen Gespräche und Hinweise sowie die vielen „Härtetests“.

Auch Dr. Beate Klöcking von der Bayerischen Landesanstalt für Wald und Forstwirtschaft, Forsthydrologie und Wasserhaushalt danke ich für die freundliche und geduldige Zusammenarbeit sowie für Hinweise bei der Erstellung und Anwendung des Präprozessingwerkzeuges.

Dagmar Haase und Gerd Schmidt und Thilo Weichel vom UFZ Leipzig-Halle bin ich für die konstruktiven Anmerkungen dankbar.

Den Mitarbeitern vom DMU in Selkeborg, vor allem Herrn Dirk Müller-Wohlfeil und Inge-Lise Madsen möchte ich für den intensiven Austausch und der damit verbundenen Abrundung vieler Funktionen beim Einsatz am Gewässernetz Dänemarks danken.

Ein sehr herzlicher Dank gilt Andreas Lechner, Andreas Löffler, Andreas Rüter, Anne-Kathrin Lindau, Daniel Wurbs, Martin Schmidt, Monika Prehn, Olaf und Sigrun Scheffler, Rainer Kurz, Stefanie Theil, Steffi Dittmer sowie meiner Mutter für das Korrekturlesen und die damit verbundene Auseinandersetzung mit einer für Außenstehende vielleicht nicht immer unterhaltsamen Materie.

Darüber hinaus gilt mein Dank natürlich all jenen, die ebenfalls bei der Entstehung dieser Arbeit direkt oder indirekt mitgewirkt haben, hier aber nicht namentlich erwähnt werden konnten.

1 Einleitung

1.1 Problemstellung und Forschungsbedarf

> *„Sustainable water resources management is a concept that emphasizes the need to consider the long-term future as well as the present. Water resource systems that are managed to satisfy the changing demands placed on them, now and on into the future, without system degradation, can be called sustainable …" [Loucks 2000, S. 65].*

Die wachsende Weltbevölkerung, ständig steigende Anforderungen an die verfügbare Menge sauberen Wassers sowie der globale Klimawandel mit Veränderungen der räumlichen und zeitlichen Muster von Niederschlag und Evapotranspiration und damit einhergehende Gefahren wie die jüngsten Hochwasserkatastrophen in Europa stellen einige der gegenwärtig bedeutendsten Herausforderungen im Umgang mit der Ressource Wasser dar [BURTON 1995]. Der größte Teil der europäischen Fließgewässer unterliegt zugleich wachsenden anthropogenen Nutzungen und ökologischen Ansprüchen. Fließgewässer sind dabei komplizierte Ökosysteme, in denen die verschiedensten Wechselwirkungen auftreten. Geogene Einflüsse, Struktur des Einzugsgebietes, Niederschläge, Temperatur, Fließgeschwindigkeit, Abflussverhältnisse, Sedimente und Substratverhältnisse sowie anthropogene Nutzungen durch Einleitungen, Entnahmen, Ausbau, Stauhaltung usw. beeinflussen ihren ökologischen Zustand. Die bestehenden wasserwirtschaftlichen und naturschutzorientierten Fragestellungen machen Zustandsbeschreibungen der betreffenden Gewässer unerlässlich. Grundlage einer qualifizierten Beurteilung dieser Ökosysteme müssen daher eine Vielzahl verschiedener hydrologischer, biologischer, physikalischer und chemischer Daten sein [SCHANZE 1998], [FUHRMANN 2001].

Einzugsgebietsmanagement ist eine Basis, um auf diese Herausforderungen zu antworten. Das Interesse an der Thematik hat dementsprechend in den letzten Jahren weltweit zugenommen [BARROW 1998]. Die DUBLIN CONFERENCE ON WATER AND THE ENVIRONMENT 1992 und die CONFERENCE ON ENVIRONMENT AND DEVELOPMENT in Rio De Janeiro 1992 seien exemplarisch für diesen Prozess genannt, dem auf europäischer Ebene vor allem die EU-Wasserrahmenrichtlinie (EU-WRRL bzw. Richtlinie 2000/60/EG) des EUROPÄISCHEN PARLAMENTes und des RATES ZUR SCHAFFUNG EINES ORDNUNGSRAHMENS FÜR MAßNAHMEN DER GEMEINSCHAFT IM BEREICH DER WASSERPOLITIK Rechnung trägt. Sie ist am 22.12.2000 im Amtsblatt der EUROPÄISCHEN GEMEINSCHAFT in Kraft getreten. Die EU-WRRL stellt die Mitglieder der EU vor neue ökologische und ökonomische Herausforderungen [INTERWIES & KRAEMER 2001]. Sie sieht Bewirtschaftungspläne für Flusseinzugsgebiete als neues Instrumentarium der Gewässerschutzpolitik vor. Kernbereiche dieser neuen Pläne sind, neben der allgemeinen Beschreibung und Abgrenzung der einzelnen Flussgebiete, die Zusammenfassung der signifikanten Belastungen, die Schutzgebietsabgrenzungen sowie Monitoring- und Maßnahmenprogramme. Hervorzuheben ist dabei die vorgesehene Einbindung der Öffentlichkeit in den Planungsprozess [FUHRMANN 2001], [BLÖCH 2001].

Nach dieser rechtsverbindlichen Grundlage ist Flussgebietsmanagement als die Gesamtheit aller Projekte und Tätigkeiten aufzufassen, die dazu dienen, die Bewirtschaftung eines Flusseinzugsgebietes nach Maßgabe der EU-WRRL durchzuführen [BLÖCH 1999], [LAWA 2000], [RENNER 2001], [DÖRHÖFER & DIBBERN 2001]. Das Hauptaugenmerk der Richtlinie liegt dabei auf:

- Bewirtschaftung von Flusseinzugsgebieten
- Erreichung bzw. Erhaltung eines „guten Zustandes" für die Gewässer

- europaweite einheitliche ökologische Qualitätsstandards
- Einbindung der Öffentlichkeit.

Mit der Verpflichtung der Mitgliedstaaten, für die Flussgebiete Bewirtschaftungspläne zu erstellen, ist die EU-WRRL eine große Herausforderung an die Wasserwirtschaftsverwaltungen der Länder, da nicht mehr in Bundesländergrenzen, sondern in Einzugsgebieten gedacht werden muss. Hier wird eine enge Zusammenarbeit zwischen den einzelnen Bundesländern erforderlich. Betrachtet man beispielsweise das Einzugsgebiet der Weser, so sind davon Bayern, Thüringen, Hessen, Niedersachsen, Nordrhein-Westfalen, Sachsen-Anhalt und Bremen betroffen. Gemäß den Vorgaben der EU-WRRL müssen in Deutschland für zehn Flussgebietseinheiten Bewirtschaftungspläne aufgestellt werden. Dabei befinden sich die Einzugsgebiete der EDER, SCHLEI/TRAVE, WARNOW/PEENE und der WESER ausschließlich in Deutschland, während alle anderen Flussgebiete (DONAU, RHEIN, MAAS, EMS, ELBE, ODER) in Kooperation mit den entsprechenden Nachbarstaaten bearbeitet werden müssen. Die Umsetzung der EU-WRRL ist auch hinsichtlich Projektsteuerung und Datenverarbeitung eine anspruchsvolle Aufgabe.

Das Fehlen optimaler Mechanismen beim Umgang mit den rasant wachsenden Datenmengen im Einzugsgebietsmanagement wird immer wieder bemängelt [THOMPSON & LAURINI 1992], [WASY 1996], [BLASCHKE 1997], [UBERTINI 1999], [USGS 2000], [LOUCKS 2002]. Gleichzeitig wird der positive Beitrag betont, den Geographische Informationssysteme (GIS) in diesem Zusammenhang leisten können [GOULTER & FORREST 1987], [BLASCHKE 1997], [FLÜGEL & STAUDENRAUSCH 1999]. In den vergangenen 30 Jahren hat sich auf dem Gebiet geographischer Informationstechnologie sowohl im öffentlichen Sektor als auch im Bereich der privaten Wirtschaft eine immense Entwicklung vollzogen [BILL 1996], [KRATZ & SUHLIG 1997] , [BLONGEWICZ 2000], [DÖRHÖFER & DIBBERN 2001]. Innovationen im Hard- und Softwarebereich ermöglichen die Anwendung dieser Konzepte in der wasserwirtschaftlichen Praxis, in der Hydrologie und nicht zuletzt auch in der hydrologischen Forschung und Modellierung, da wasserwirtschaftliche Problemstellungen raumbezogen und somit auf Punkte, Linien und Flächen als die Geometrieelemente eines GIS bezogen sind [PFÜTZNER ET AL. 1999]. GIS-basierten Prognose- und Managementsystemen wird eine zunehmend größere Bedeutung bei der Unterstützung von Entscheidungsprozessen in der räumlichen Planung beigemessen [DUTTMANN & HERZIG 2002].

Dazu trägt auch das Konzept räumlicher Datenbanken bei, in denen eine relationale (z. B. MSACCESS™, ORACLE™), objekt-relationale (z. B. POSTGRESQL™) oder objektorientierte Datenbank (OBJECTSTORE™, POET™) um eine räumliche Komponente (z. B. SDE™ auf ORACLE, POSTGIS© auf PostGreSQL) erweitert wird. Dieser Weg führt in einen Bereich nahezu unbegrenzten Potenzials hinsichtlich des Datenmanagements [EGENHOFER 1994], [PAIVA 1998], [DJAFRI ET AL. 2002], [GRIFFITHS ET AL. 2002].

Um diesen verschiedenen Anforderungen und Möglichkeiten gerecht zu werden, beschäftigte sich die Arbeit auch damit, *„... to combine GIS with more increasingly sophisticated databases ... and the Internet...*" [CAR & TAYLOR 1999 S.3]. Die Konvergenz unterschiedlicher Technologien sollte dabei nicht nur High-End-Produkte führender Vertreter der kommerziellen Entwicklung dieses Bereiches einschließen (gedacht sei dabei z. B. an ARCGIS™, ORACLE™ oder INFORMIX™), sondern auch Wege zur Integration möglichst weit verbreiteter und preiswerter (im Idealfall Opensource-) Anwendungen aufzeigen [BURTON 1995], [OGC 2001]. An dieser Schnittstelle setzt die Arbeit an und verfolgt dabei das

Ziel, mit der Abbildung von Hierarchien zur Optimierung von Einzugsgebietsmanagement beizutragen.

Zentraler räumlicher Betrachtungsgegenstand der Arbeit ist das oberirdische Einzugsgebiet der SALZA (**Abb. 13**), einem linken Nebenfluss der SAALE, *„... der sich, bezogen auf seine ökologische Funktion, bisher nur in wenigen Abschnitten als intakt bezeichnen lässt ...“* [SCHANZE 1998, S. 6]. Im Interesse der Sicherung der Mansfelder Seen als Bestandteil des Naturhaushalts wurde durch das Regierungspräsidium Halle (ehemals Staatliches Amt für Umweltschutz - STAU) der „Bewirtschaftungsplan Salza“ (1995 bis 2002) aufgestellt. Er dient der Ordnung der Nutzungen des Gewässerregimes in ihrem Einzugsgebiet, der Zusammenführung der Daten zur biologischen, chemisch-physikalischen und morphologischen Gewässerbeschaffenheit hinsichtlich geeigneter Bewirtschaftungsmaßnahmen und deren Bewertung auf Basis eines parameterbezogenen Zielsystems zur angestrebten Umweltqualität. Die im Rahmen dieses Bewirtschaftungsplanes und der standardmäßigen Kontrolle der Gewässerbeschaffenheit erhobenen Daten liegen in zeitlich und räumlich hoher Dichte vor.

Eine optimale Planung, Realisierung und Kontrolle von Aufgaben im Gewässerschutz erfordert die Auseinandersetzung mit den topographischen Beziehungen innerhalb des betrachteten Einzugsgebietes. *„Wesentliche Grundlage wasserwirtschaftlichen Planens und Handelns ist die Erfassung, Verarbeitung, Aktualisierung und zeitnahe Auswertung von Daten vielfältigen Ursprungs, ... die den Fließgewässern und ihren Abschnitten zugeordnet werden müssen...“* [LAWA 1978, S. 5]. Diese Beziehungen sind morphologisch durch Parameter wie Flussdichte, Flusshäufigkeit oder Verzweigungsverhältnis charakterisiert und folgen in ihrer primären Ausprägung natürlichen Rahmenbedingungen wie Geologie, Klima und Vegetation [BLASCHKE 1997]. Sie führen in Mitteleuropa natürlicherweise im Allgemeinen zu baumstrukturierten Gewässernetzen [MARCINEK & ROSENKRANZ 1989], deren hierarchische Zusammenhänge im Umgang mit gewässerbezogenen Daten durch diese Arbeit besser nutzbar gemacht werden sollen. Dabei wurden auch Sonderfälle wie z. B. Kreisstrukturen oder Deltas untersucht und ggf. implementiert.

Hierarchien werden in den meisten Datenbanksystemen (DBMS) bisher nicht explizit unterstützt. 1:n-Beziehungen können über Primär-/Fremdschlüssel-Konstruktionen abgebildet werden, dies erfordert jedoch die Reflexion der hierarchischen Beziehungen in Metadaten [HEUER 1997]. Ziel war es, den Sprachschatz von Standard-SQL (Structured Query Language) für diese Zwecke auszunutzen, ohne auf spezielle, DBMS-abhängige Erweiterungen angewiesen zu sein. SQL ist die offizielle Standardsprache für die Etablierung einer Schnittstelle zu relationalen Datenbanken. Dies hat Vorteile in der relativ leichten Übertragbarkeit des Verfahrens auf unterschiedliche Datenbank-Produkte und lässt sich, gering modifiziert, auch innerhalb der meisten GIS anwenden [LEHNER 1998], [FLÜGEL & STAUDENRAUSCH 1999].

„The stream ordering processes may become a not difficult but time consuming process especially on large basins, large scale studies and dense networks. The use of Vector GIS methodologies to deal with stream network studies introduces the possibility of making use of computer facilities to solve the question” [CAR & TAYLOR 1999, S. 9]. Ein weiteres wichtiges Ziel der Arbeit bestand darin, geeignete Wege zu finden, die Struktur eines Gewässernetzes automatisiert zu analysieren, zu speichern und zu nutzen. Eine Automatisierung bedeutet dabei den Entwurf und die Implementierung von Programmen zur Analyse der Gewässerhierarchie und ihre Fixierung in Form von Metadaten sowie ihrer Nut-

zung aus der Datenbank und dem GIS heraus. Die Erfassung der Gewässerstruktur, die Hierarchisierung, aus der reinen geometrischen Information – also ohne die Zugrundelegung von nicht immer vorhandenen bzw. nicht in jedem Gelände bedenkenlos einsetzbaren Geländemodellen (vgl. [HEUVELINK 1993, KUMLER 1994], [BÜYÜKSALIH & JACOBSEN 2002]) – erfordert, neben dem bereinigten Gewässernetz, eine Auseinandersetzung mit Möglichkeiten, die räumlichen Verhältnisse mithilfe geeigneter Algorithmen zu erfassen und abzubilden. Die so entstandenen Werkzeuge sollen die Ergebnisse der Arbeit auch auf andere Gebiete übertragbar machen[1].

Leider befinden sich digitale Geometriedaten nicht immer in einem für solche Automationen geeigneten Zustand. Digitalisierung und Weiterverarbeitung lassen Raum für verschiedene Fehlerquellen wie z. B. nicht geschlossene Abschnittsenden, falsche Digitalisierungsrichtung, Überschneidungen, Überlagerungen usw. Am Anfang der Arbeit stand daher die Suche nach Möglichkeiten der weitgehend automatisierten Aufbereitung von Gewässernetzen und deren Realisierung in Programmansätzen. Diese Automatisierung und die sich daraus ergebenden Möglichkeiten zur Standardisierung soll auch den grenzüberschreitenden Datenaustausch zwischen den EU-Ländern unterstützen und damit verbundene Synergieeffekte verstärken.

Unter dem Aspekt enger werdender gesetzlicher Regelungen erlangen computerbasierte Instrumente zur Erleichterung von Entscheidungsfindungen (Decision Support Systems) immer größere Bedeutung und sind in den letzten Jahren zu einem wesentlichen Bestandteil der Schnittstelle zwischen Wissenschaft und Politik geworden. Ergebnisse der Arbeit wurden daher in anwenderfreundlichen GIS-Applikationen exemplarisch zusammengeführt, die mit ihren Funktionalitäten zur Aufbereitung, topologischen Analyse und Hierarchisierung von Gewässernetzen und mit Abfrage- und Managementoptionen für das Untersuchungsgebiet die Bandbreite des Einsatzes der Ergebnisse in der Praxis aufzeigen sollen. Die effiziente Datenhaltung innerhalb eines Relationalen Datenbank-Management-Systems (RDBMS) kann, zusammen mit einer optimalen Abbildung der Gewässerstruktur, die Effizienz von Gewässermonitoring oder anderer Aspekte der Oberflächengewässerbewirtschaftung erhöhen. Einige dieser Möglichkeiten sollen hier verdeutlicht werden. In diesem Zusammenhang werden Fragen und Aufgabenstellungen wie die folgenden bearbeitet und Modellansätze skizziert:

- In welcher Zeit legt Stoff X den Weg von Abschnitt A zu Abschnitt B zurück? Wie lang ist die Strecke zwischen diesen Abschnitten? Wie viele Abschnitte, welche Arten von Knoten und welche Messstellen befinden sich auf diesem Weg?
- Welche Menge eines Stoffes X fließt (kumulativ) in einen bestimmten Punkt? Welche Ortschaften, Zusammenflüsse und Quellen liegen im Einzugsgebiet dieses Punktes?
- Welche Flussdichte bzw. Flusshäufigkeit und welches Verzweigungsverhältnis finde ich oberhalb einer Messstelle? Wie verteilen sich STRAHLER-, SHREVE-, HORTON- oder die Klassische Flussordnung in ihrem Einzugsgebiet? Ist dieses Einzugsgebiet streng hierarchisch oder befinden sich Kreisstrukturen im Gewässernetz?
- Welche Abschnitte gehören zum Hauptfluss eines Gewässernetzes?
- An einem bestimmten Abschnitt traten erhöhte Messwerte auf. Wie ist die Verteilung des Inputs an den oberhalb liegenden Messstellen? Welches sind die Hauptquellen bzw. welche Werte dieses Teils des Messnetzes weichen vom jeweiligen Durchschnitt oder einem festgelegten Zielwert ab?

[1] Alle Algorithmen und Programme sind, sofern nicht anders beschrieben, durch den Bearbeiter entwickelt worden.

- Wie schnell breitet sich eine Hochwasserwelle in den oberhalb einer Ortschaft liegenden Abschnitten aus? Welches ist das potentiell von einer Überschwemmung erfasste Gebiet?
- Ist es möglich, dass eine Fischpopulation in der Zeit t den Weg von Abschnitt A zu Abschnitt B (auch stromab $\Rightarrow$ stromauf) zurücklegt?
- Wo liegen Stationierungsachsen bzw. die Marken einer Kilometrierung auf Basis eines vorgegebenen Intervalls innerhalb des Gewässernetzes?

Besonders hinsichtlich der EU-Wasserrahmenrichtlinie könnte solch ein hierarchiebasiertes Gewässermanagementsystem unterstützende Funktionen übernehmen. Damit verbundene Anfragen an den Datenbestand sollten möglichst weitgehend mit Unterstützung von SQL formulierbar sein. Aufgrund der weiten Verbreitung in Behörden, wissenschaftlichen Einrichtungen und der privaten Wirtschaft wurden als Basis der Entwicklung das GIS ARCVIEW™ sowie MAPOBJECTS™ mit VISUALBASIC™ gewählt. Parallel dazu sind Teilaspekte auch internetfähig mit PHP©, MySQL© und einem UMN-MAPSERVER© realisiert worden.

Die Analyse der verfügbaren Literatur zeigt, dass zwar eine Reihe von Konzepten existieren, um aus der reinen Geometrie digitaler Fließgewässernetze hierarchische Attribute datenbanknutzbar abzuleiten, wobei das Instrumentarium GIS sehr hilfreich sein kann [FINDEISEN 1990], [LEHNER 1998], [VERDIN & VERDIN 2000]. Alle diese Verfahren haben jedoch hinsichtlich der Effektivität ihrer Anwendung sowohl im GIS- als auch im Datenbankbereich ihre spezifischen Vor- und Nachteile. Hier setzte diese Arbeit an, versuchte, geeignetere Methoden zu entwickeln und ihre Ergebnisse unter dem Aspekt der Transformierbarkeit auf beliebig große bzw. komplexe Gewässernetze ausreichend allgemein zu halten.

1.2 Problembezogene Fragestellungen und Arbeitshypothesen

Aus den vorangegangenen Abschnitten zeichnet sich eine deutliche Dreiteilung der Aufgabenstellung ab, für die folgende Fragestellungen abzuleiten sind:

Präprozessing

- Welche Bedingungen muss ein digital vorliegendes Fließgewässernetz erfüllen, um hierarchisiert werden zu können?
- Wie lassen sich diese Bedingungen weitestgehend automatisch und nutzerfreundlich schaffen?

Hierarchisierung

- Wozu dienen hierarchiebezogene Abfragen? Welche Eigenschaften müssen hierarchische Attribute aufweisen, um sie möglichst effektiv in einer Datenbank und einem GIS abzubilden?
- Welche Typen der Hierarchisierung bestehen bereits und inwiefern sind sie geeignet, sowohl im RDBMS als auch im GIS verwendet zu werden? Was sind die jeweiligen Vorteile?
- Wie müssten SQL-Abfragen an die Datenbank formuliert werden?
- Gibt es Alternativen zu SQL-Abfragen und wie könnten alternative Wege der Reflexion von Gewässerstrukturen aussehen? Inwiefern lassen sich diese umsetzen und anwenden?
- Wie lassen sich hierarchische Informationen aus der bereinigten Geometrie automatisiert extrahieren und in Form von Attributen abbilden?

- Welche Voraussetzungen sind nötig, um die entstehenden Metadaten auch auf andere räumliche Objekte wie Ortschaften, Messnetze oder Teileinzugsgebiete zu übertragen und in diesen Datenbeständen zu nutzen?

Fließgewässermonitoring / Datennutzung

- In welcher Form sind die gewonnenen Informationen am hilfreichsten?
- Inwiefern lassen sie sich in Kombination bzw. getrennt, im GIS als auch im RDBMS einbinden?
- Welche Ordnungssysteme sind für die unterschiedlichen Fragestellungen zum Fließgewässermanagement in welcher Weise geeignet?

Diese Fragen führen zu folgenden **Annahmen**:

Präprozessing

- Grundsätzlich lässt sich jedes digital vorliegende baumstrukturierte Gewässernetz in einen für die automatisierte Hierarchisierung geeigneten Zustand überführen[2].
- Es gibt Möglichkeiten, diesen Prozess mithilfe programmierter Funktionen quantitativ, qualitativ und zeitlich zu optimieren. Hier muss die Suche bzw. Erstellung solcher Abläufe im Vordergrund stehen.
- Diese Abläufe sollen weitgehend allgemeingültig sein. Daher sollten sie an unterschiedlichen Datengrundlagen entwickelt, validiert und optimiert werden.

Hierarchisierung

- Die hierarchischen Beziehungen jedes normalhierarchischen Fließgewässernetzes, so auch das der SALZA, lassen sich automatisiert extrahieren und als Attribute abbilden.
- Der Prozess der Hierarchisierung ist allgemeingültig und grundsätzlich mit verschiedenen GIS-Systemen durchführbar. Daher wurde auch dieser Arbeitsschritt an unterschiedlichen Systemen (ARCVIEW, MAPOBJECTS) entwickelt.
- Die Abbildungen von Hierarchien sind grundsätzlich auch auf andere räumliche Objekte wie Ortschaften, Messnetze oder Teileinzugsgebiete übertragbar. Auch hier gilt es, automatisierte Prozesse zu erstellen und an unterschiedlichen Testgebieten zu validieren.

Fließgewässermanagement / Datennutzung

- Die Abbildung der Fließgewässerhierarchien in Metadaten kann Fließgewässermanagement optimieren.
- Diese Ordnungssysteme lassen sich in hierarchieorientierten SQL-Abfragen sowohl im GIS als auch im RDBMS – gelinkt und separat – anwenden. Sie sind in leicht verfügbaren Softwarelösungen (ARCVIEW, MSACCESS, MySQL) wie in „High-End-Produkten“ (ArcGIS, MAPOBJECTS mit ArcSDE und ORACLE) implementierbar. Die Umsetzung erfolgte daher mit ARCVIEW / MSACCESS, MAPOBJECTS / ORACLE / ArcSDE sowie in einer Internetanwendung innerhalb einer Drei-Schicht-Architektur aus MySQL, Mapserver, APACHE-Server und HTML (PHP)-Client.

[2] Die dabei geforderten und entstehenden Eigenschaften werden in **Kap. 2** näher erläutert.

1.3 Allgemeine Zielstellungen

Die Ziele der Arbeit bestehen zusammengefasst in folgenden Punkten, aus denen sie zugleich ihre Praxisrelevanz bezieht:

- Fehler in der geometrischen Datenbasis (z. B. ATKIS) sollen möglichst weitgehend automatisiert und nutzerfreundlich analysiert und behoben werden können.
- Bereits vorhandene Flussordnungskonzepte sollen auf ihre Verwendbarkeit hinsichtlich der genannten Fragestellungen untersucht bzw. besser einsetzbare Methoden zur Erfassung und Abbildung von Fließgewässerhierarchien implementiert werden. Dazu gilt es zunächst, die zugrunde liegenden Kriterien herauszustellen.
- Die große Menge oftmals bereits vorhandener Daten muss in einen solchen hierarchiebasierten Ansatz integrierbar und die resultierenden hierarchischen SQL-Statements mit dem ANSI-SQL-Standard umsetzbar sein.
- Fließgewässermanagement soll für einen möglichst großen Kreis von Anwendern optimierbar werden. Der unterschiedlichen Ausstattung mit Soft- und Hardware in Ämtern und Institutionen sollen möglichst effiziente, allgemeingültige und übertragbare Abbildungen von Baumstrukturen hinsichtlich einer besseren Verfügbarkeit und Ausnutzung vorhandener Daten Rechung getragen werden.
- Auch die Entkopplung von GIS und Datenbank bezüglich der Anbindung hierarchischer Attribute im Zeichen optimaler Einsatzes der Ergebnisse dieser Arbeit wird angestrebt. Dies soll an unterschiedlichen Anwendungen gezeigt werden.
- Der Forderung der EU-WRRL nach Veröffentlichung des Gewässerzustandes soll eine exemplarische Internet-Anwendung entsprechen, die auf einer Datenbank, einer GIS-Komponente sowie den hierarchischen Metadaten der verschiedenen gewässerbezogenen Datenschichten basiert.

Die Arbeit wurde in Konzeption und Umsetzung nach diesen Vorüberlegungen ausgerichtet. **Abb. 1** zeigt als ersten Überblick zum Inhalt der Arbeit das daraus abgeleitetet Verarbeitungs- und Nutzungsschema digitaler Fließgewässernetze und darauf bezogener Daten. Hier spiegelt sich ebenfalls die deutlichen Dreiteilung der Aufgabenstellung und der damit verbundenen Problemlösungskomplexe wider. Die weiteren Ausführungen werden diesem Schema folgen.

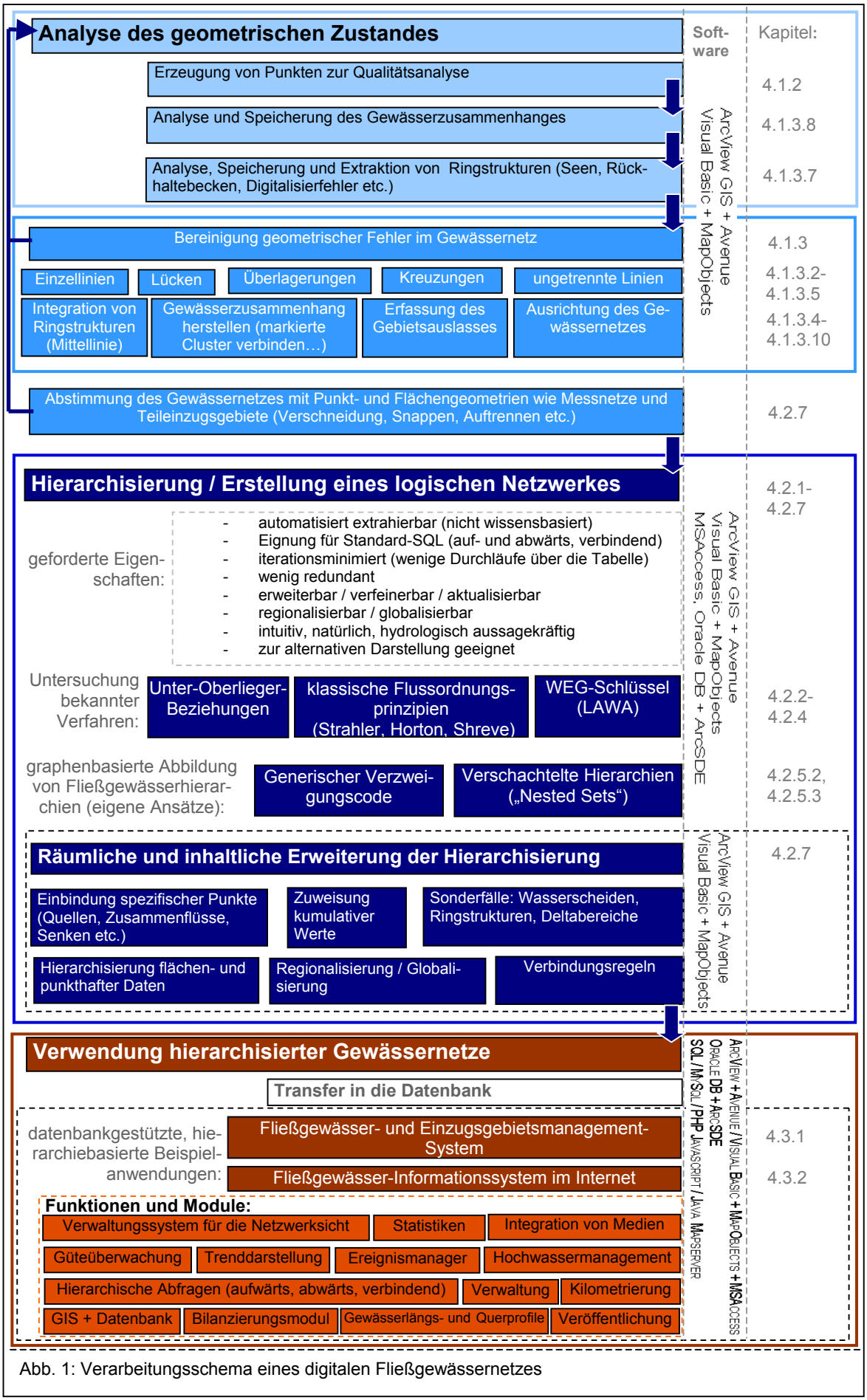


Abb. 1: Verarbeitungsschema eines digitalen Fließgewässernetzes

2 Inhaltliche und methodische Ansätze

"All these prepositions are empirical. It is not possible to prove them mathematically. In fact, it is a rather simple matter to demonstrate by rational hydraulic analysis that not a single one of them is mathematically accurate. Fortunately, nature is not aware of this …" [JOHNSTONE & CROSS 1949 In: VAN DEURSEN 1995, S. 64].

2.1 Abflussbildung, Abflusskonzentration und Gewässerstrukturierung

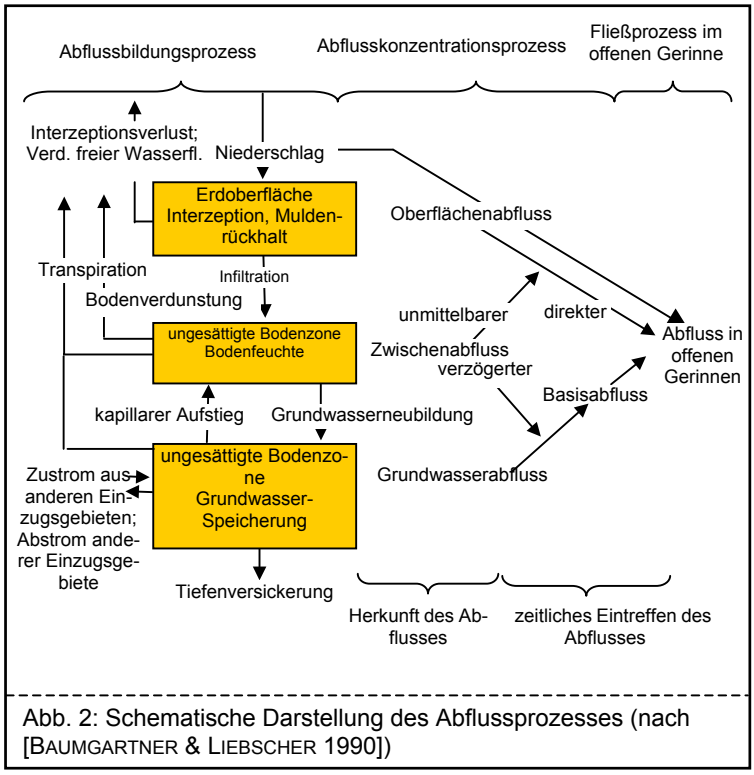


Abb. 2: Schematische Darstellung des Abflussprozesses (nach [BAUMGARTNER & LIEBSCHER 1990])

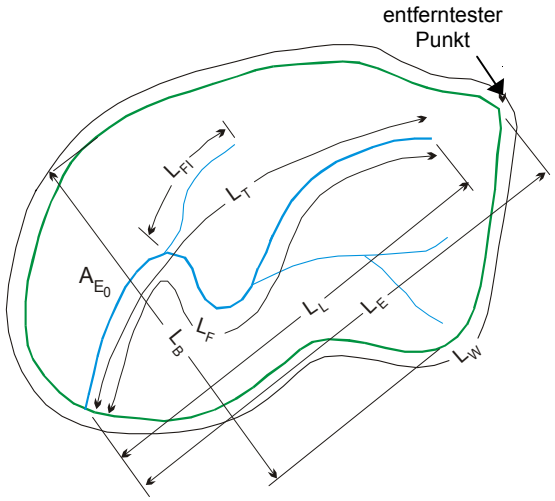
Bei der Abflussbildung wird der Anteil des Niederschlages, der den Erdboden erreicht (abflusswirksamer oder effektiver Niederschlag) entweder auf der Oberfläche in die nächstgelegenen Fließgewässer transportiert, oder versickert im Boden und der wasserungesättigten Zone. Die Aufnahme- und Wasserleitfähigkeit des Untergrundes hängen von den jeweiligen Wasserverhältnissen und den Eigenschaften der geologischen Schichten ab. Ist die Niederschlagsintensität größer als die Infiltrationsrate, kommt es zum Oberflächenabfluss. Dieser kann sich wiederum aufgliedern in einen Teil, der direkt in die Gewässer gelangt, und einen Teil, der sich in Mulden auf der Oberfläche sammelt und verzögert infiltriert bzw. der Verdunstung unterliegt. Bei Fortsetzung der Infiltration gelangt das nunmehr unterirdisch fließende Wasser aus den oberen Bodenhorizonten durch die ungesättigte Zone entweder ins Grundwasser, sorgt dort für die Grundwasserneubildung und erreicht mit dem Grundwasserfluss und erheblicher Verzögerung die oberirdischen Gewässer, oder es strömt als Zwischenabfluss (Interflow) unterirdisch den Wasserläufen zu bzw. bildet Wasserströmungen aus zeitweilig gesättigten Schichten, die sich durch die oberen Schichten einer Formation mit einer Menge bewegen, die weit über der normalen Versickerung liegt. Ein geringer Teil kann auch in tiefere Schichten versickern (**Abb. 2**) [BAUMGARTNER & LIEBSCHER 1990].

Flüsse entwickeln sich auf geneigten Flächen überall dort, wo das Niederschlagswasser nicht durch Verdunstung und zeitweilige Versickerung aufgebraucht wird. Sie *„... sind somit in langgestreckten, einseitig geöffneten Hohlformen der Landoberfläche fließende natürliche Wasserläufe, die eine umgrenzbare Fläche des Festlandes mit natürlichem Gefälle entwässern."* [MARCINEK & ROSENKRANZ 1989, S. 154]. Diese Fläche ist das oberirdische Einzugsgebiet (**Abb. 3**), die Umgrenzung die Wasserscheide und der Prozess die Abflusskonzentration. Diese Abflusskonzentration hängt in ihrem zeitlichen Verlauf und ihrer quantitativen Ausprägung von der Beschaffenheit der Landoberfläche und des Untergrundes ab [WILHELM 1993]. Sie wird wesentlich bestimmt durch

- die Größe des Einzugsgebietes A_{EO}

- das Gefälle J
- den längsten Laufweg l im Einzugsgebiet
- sowie weitere morphometrische und hydraulische Parameter.

Abflussbildung, Abflusskonzentration und Durchflussentwicklung führen zum Abfluss Q, einem Wasservolumen, welches pro Zeiteinheit einen definierten oberirdischen Quer-schnitt durchfließt (Maßeinheit: m^3/s oder l/s). Dieser Abfluss setzt sich aus verschiedenen Komponenten atmosphärischen und terrestrischen Ursprungs zusammen. Der Einzugsgebietskörper reicht dabei von der äußersten Grenzlinie der Einzugsgebietsfläche bis zur tiefsten Sohle des noch an der Abflussbildung beteiligten Grundwasserleiters (**Abb. 4**). Die oberirdische Wasserscheide begrenzt nur das oberirdische Einzugsgebiet, während für den unterirdischen Abfluss das unterirdische Einzugsgebiet maßgebend ist. Beide Gebiete können voneinander abweichen (**Abb. 4**), wobei die Unterschiede vom Betrachtungsmaßstab sowie den geographischen und hydrogeologischen Gegebenheiten abhängen [DYCK & PESCHKE 1995]. Ist der Abfluss auf die zugehörige Einzugsgebietsfläche bezogen, so wird er als Abflussspende q ($l/(s*km^2)$) oder als Abflusshöhe h_A (mm/Zeiteinheit) bezeichnet [BAUMGARTNER & LIEBSCHER 1990].



L_F=Flusslänge, l_T=Tallänge, l_L=Luftlinie, l_E=Länge des Einzugsgebietes, l_B=Breite des Einzugsgebietes, l_W=Länge der oberirdischen Wasserscheide, A_{EO}=Fläche des oberirdischen Einzugsgebietes

Abb. 3: Schematische Darstellung eines Einzugsgebietes (nach [DYCK & PESCHKE 1995])

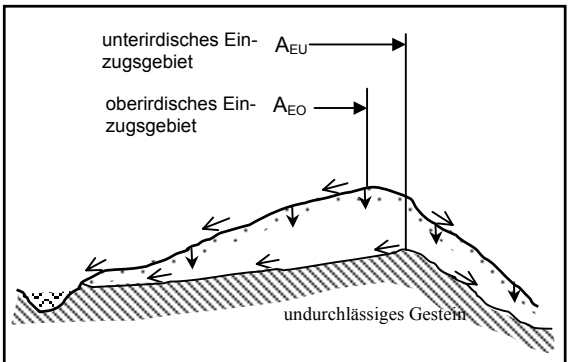


Abb. 4: Ober- und unterirdisches Einzugsgebiet (nach [WOHLRAB ET AL. 1992])

Zur Beschreibung der Form eines Einzugsgebietes dienen neben der Fläche A_{EO} auch die Einzugsgebietslänge L_E und die Einzugsgebietsbreite L_B (**Abb. 3**). Für die Beschreibung der Gebietsform ist der Formfaktor nach HORTON [HORTON 1932] der am häufigsten gebrauchte: $f_H = A_{EO}/l_E^2$. Er beeinflusst die Abflusskonzentration und die Laufzeit und lässt sich als : $f_H = A_{EO}/l^2$ darstellen[3] (**Abb. 3**).

Im Unterschied zum globalen Wasserkreislauf erfährt der hydrologische Kreislauf eines Flusseinzugsgebietes sowohl Input (Niederschlag) als auch Output (Abfluss). Mit dem Niederschlagswasser werden dem System Feststoffpartikel, Säuren und Salze zugeführt. Dabei produziert das Einzugsgebiet selbst organische Substanz, die gemeinsam mit anorganischem Material als Sediment aus dem System abgeführt werden kann. Dieses System lässt sich, bezogen auf die Wasserein- und ausgabe in der Wasserhaushaltsglei-

[1] l ist der längste Laufweg des Wassers vom entferntesten Punkt des oberirdischen Einzugsgebietes bis zum Gebietsauslass.

chung ausdrücken: N = A + V + (R-B) (N = Niederschlag, A = Abfluss, V = Verdunstung, R = Rücklage (Speicher), B = Aufbrauch) [MARCINEK & ROSENKRANZ 1989]. Als Speicher fungieren die Vegetation, der Boden, der Gesteinsuntergrund sowie Standgewässer. Die Ausstattung eines Einzugsgebietes mit Speichern übt einen entscheidenden Einfluss auf dessen Abflussverhalten aus. Rücklage und Aufbrauch gestalten sich im Allgemeinen längerfristig ausgeglichen [SCHMIDT 1984]. Dabei sind jedoch die Werte der drei Hauptkomponenten der Wasserhaushaltsgleichung Niederschlag, Abfluss und Verdunstung für jedes Einzugsgebiet unterschiedlich. Die Fließzeiten des Zwischenabflusses sind wesentlich länger als die des Landoberflächenabflusses, während der Basisabfluss über das Grundwasser im allgemeinen am längsten dauert [BAUMGARTNER & LIEBSCHER 1990], [WILHELM 1993].

Der Abflussprozess beruht im wesentlichen auf drei unterschiedlichen, meist jedoch gleichzeitig ablaufenden Vorgängen[4]:

- der Abflussbildung aus dem Niederschlag
- der Konzentration des zum Abfluss gelangenden Niederschlages und
- dem Fließprozess im offenen Gerinne.

Das durch den Abfluss genährte Gewässernetz ist ein in sich verbundenes, verzweigtes System in Form von Bächen und Flüssen [MARCINEK & ROSENKRANZ 1989]. Abflusskonzentration, Landoberfläche und Untergrund bestimmen die hierarchische Anordnung innerhalb eines Fließgewässernetzes. Quellen sind generell in den höheren Bereichen der Einzugsgebiete, nahe der Wasserscheide zu finden. Jedes Einzugsgebiet besteht aus einem Netzwerk von Flussabschnitten und sammelt sich im jeweils tiefsten Punkt dieser Einheit, um ggf. einem übergeordneten System (aus mehreren Einzugsgebieten) zuzuströmen. Am Ende fließt das Gesamtsystem auf einen einzigen Punkt zu, in der Regel in das Meer oder einen Endsee [MARCINEK & ROSENKRANZ 1989], [WILHELM 1993]. Die räumlichen Dimensionen von Flusseinzugsgebieten erstrecken sich dabei von Hangfacetten bis zu kontinentalen Flusseinzugsgebieten. Entlang des Gewässers reicht die Betrachtungsweise von kurzen Segmenten über längere Abschnitte bis hin zur Gesamtlänge eines Flusses.

2.2 Hierarchieprinzip und Baumstruktur

Das Hierarchieprinzip ist ein formales System, bei dem jedes Element, bis auf das höchste, ein übergeordnetes Element und bis auf das jeweils unterste, ein oder mehrere untergeordnete Elemente haben kann. Es wird exakt durch die Baumstruktur dargestellt und spielt in Computersystemen eine wichtige Rolle, weil sie klar und übersichtliche Strukturen bewirkt wie z. B. bei der Steuerungshierarchie, in der strukturierten Programmierung oder bei Datenverzeichnissen [SCHULZE 2000].

Eine Baumstruktur entsteht durch die logische Gliederung einer Gesamtheit in Teilelemente, wobei man diese Gliederung auf unteren Stufen beliebig fortsetzen kann. Der Ausgangspunkt wird als Wurzel bezeichnet, die davon ausgehenden Objekte als Zweige oder Knoten. Gliedert sich ein Zweig nicht weiter nach unten, so bezeichnet man ihn als Blatt oder Quelle. Einen Zweig, der nicht Wurzel ist, bezeichnet man mit seinen Nachfolgern als Teilbaum oder Cluster (Typ 5 in **Abb. 5** weist mehrere Cluster auf). Aus dieser Struktur ergibt sich ein Graph (vgl. **Kap. 4.2.5**), der einem mit der Wurzel nach oben ste-

[4] Diese lassen sich in weitere, verschiedenen Gesetzen und Abhängigkeiten folgende Unterprozesse gliedern.

henden Baum ähnelt. Man unterscheidet normale Bäume (mit beliebig vielen Zweigen auf jeder Ebene) und Binärbäume (mit jeweils nur zwei Zweigen pro Ebene). Normalhierarchische Gewässersysteme entsprechen solchen Binärbäumen. Die Baumstruktur lässt sich für unterschiedliche Zwecke verwenden, so z. B. zur strukturierten Programmierung und Systementwicklung, für die Darstellung von Dateiverzeichnissen und bei komplexen Entscheidungsprozessen, wie sie im Fließgewässermanagement die Regel sind [BLASCHKE 1997]. In Fließgewässern können so die Wasser- und Stoffflüsse sowie aquatische Habitatbeziehungen nachvollzogen und simuliert werden [BLONGEWICZ 2000].

Von den in **Abb. 5** skizzierten Fließgewässertypen entsprechen demnach alle diesen Kriterien, wenngleich die Typen 1,4,5 und 7 keine Binärbäume darstellen. Dort rot umrandete

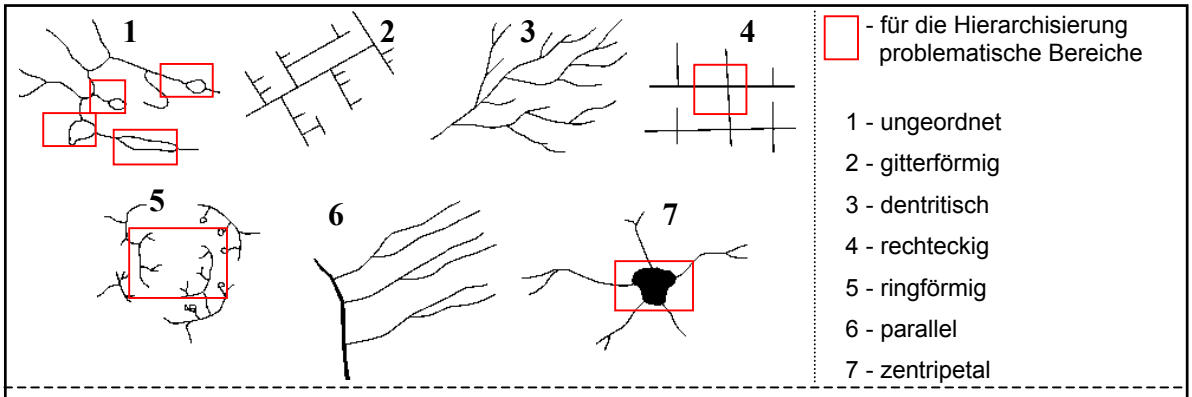


Abb. 5: Fließgewässertypen (nach [MARCINEK, & ROSENKRANZ 1996])

Problembereiche werden in **Kapitel 4.1** behandelt.

2.3 Überblick zu vorhandenen Flussordnungskonzepten

Form und Funktion eines geomorphologischen Systems sind das Produkt verschiedener, in einem weiten Bereich zeitlicher und räumlicher Variabilität wirkender Prozesse. *„The hierarchical structure of geomorphologic systems such as the drainage network has been a traditional challenge for research …"* [VAN DEURSEN & KWADIJK 1990, S. 254]. Das Verständnis eines Einzugsgebietes als Wasser-, Energie- und Stoffverteilungssystem kann durch eine interne numerische Organisation wesentlich erleichtert werden, wodurch in den vergangenen Jahrzehnten eine Reihe von Kodierungssystemen für Gewässer und deren Einzugsgebiete von verschiedenen Organisationen und Institutionen erstellt wurden [VERDIN & VERDIN 2000]. Sie folgen jedoch unterschiedlichen Voraussetzungen und Zielstellungen. Zu den ältesten und verbreitetsten Flussordnungssystemen zählen das von HORTON [HORTON 1945], STRAHLER [STRAHLER 1964], SHREVE [SHREVE 1967] und die Klassische Flussordnung (**Abb. 6**).

In der Klassischen Flussordnung wird die Eins dem Hauptfluss zugeschrieben. Alle in diesen mündenden Flüsse erhalten die zweite Ordnung usw. (**Abb. 6**). Die Gewässer Deutschlands wurden beispielsweise nach diesem System organisiert, wobei die Funktion eines Flusses (z. B. als Wasserstrasse) und einige andere Aspekte von Bedeutung waren [LEHNER 1998].

Die SHREVE Link Magnitude addiert die Anzahl von Zuflüssen erster Ordnung (Quellen) oberhalb eines betrachteten Abschnittes und kann somit als (Schätz-) Maß für das Was-

servolumen oder andere quantitative Werte herangezogen werden (**Abb. 6**) [CHENG ET AL 2001].

Beim STRAHLER-System bilden zwei Flüsse erster Ordnung eine zweite Ordnung etc. Es wird oft für hydromorphologische Analysen innerhalb von Wasserscheiden genutzt.

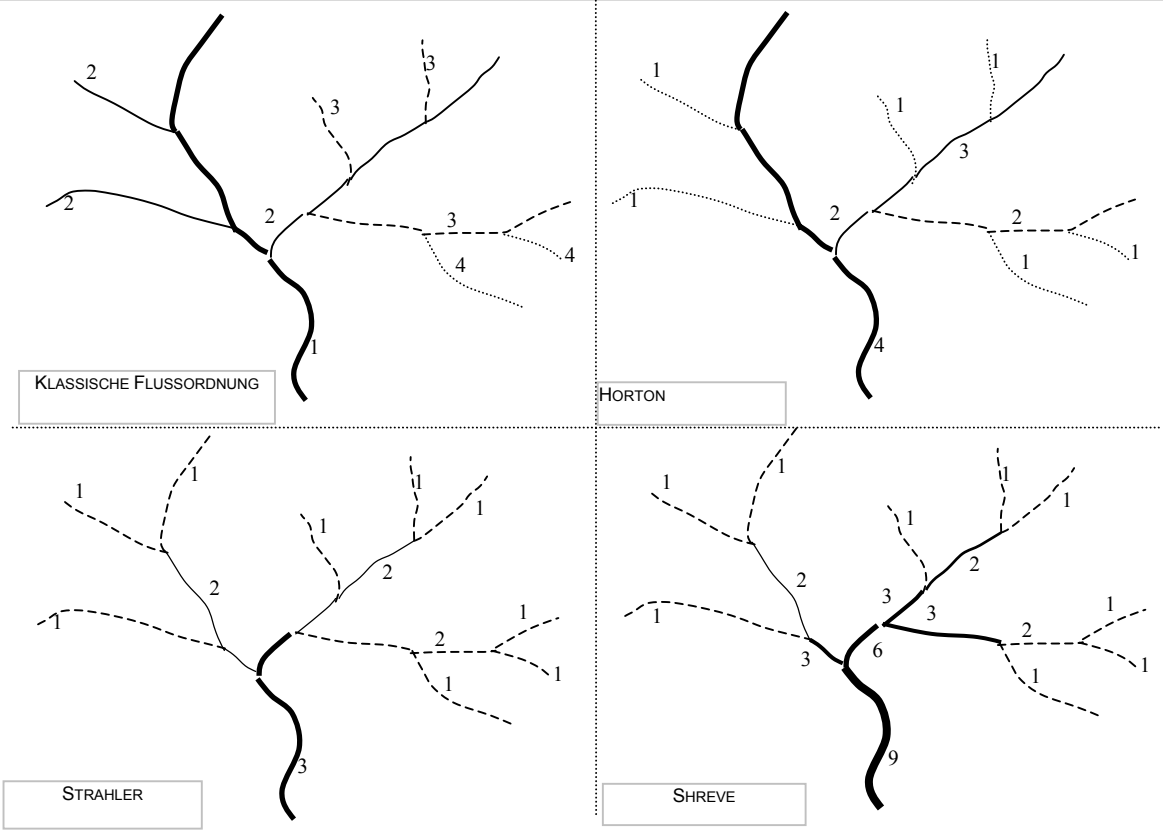


Abb. 6: Klassische Flussordnungskonzepte [nach BAUMGARTNER & LIEBSCHER 1996]

Die LÄNDERARBEITSGEMEINSCHAFT WASSER (LAWA) schlug 1970 in der „Richtlinie für Gebietsbezeichnungen“ eine bundeseinheitliche Systematik zur Bezeichnung von Einzugsgebieten oder darin gebildeter Teileinzugsgebiete vor. Die Gebietskennzeichnung erfolgt durch zehnstellige Codes, die eine numerische Verschlüsselung der oberirdischen Einzugsgebiete Deutschlands darstellen und eine Zuordnung von Objekten und Daten ermöglichen. Da jedoch diese Zuweisung auf Basis des Fach- und Regionalwissens der Bearbeiter erfolgt, ist sie nicht automatisierbar. Stromauf- und abwärtsgerichtete SQL-Statements lassen sich nur mit relativ komplexen Stringoperationen formulieren. Dennoch wurde sie wegen ihrer nationalen Bedeutung in dieser Arbeit näher untersucht, beschrieben und implementiert (vgl. **Kap. 4.2.3**).

Das HYDROLOGIC UNIT SYSTEM der WATER RESOURCES DIVISION des US GEOLOGICAL SURVEY (USGS) unterteilt das Territorium der USA in 21 Haupt- und 222 Subregionen [SEABER ET AL. 1987]. Diese Subregionen wiederum sind in sukzessive kleinere Gebiete unterteilt. Der resultierende HYDROLOGIC UNIT CODE besteht aus acht Stellen, von denen jeweils zwei die Region, die Subregion, die Verwaltungseinheit und eine Unter-Verwaltungseinheit repräsentieren. Die USGS und anderen föderale und staatliche Einrichtungen nutzen dieses System, es wurde bisher jedoch nicht auf Gebiete außerhalb der USA übertragen.

Das NATIONAL WATER INFORMATION SYSTEM (NWIS) der USA ist die Basis für USGS - Wassermessungen. NWIS beinhaltet ein System achtstelliger, stromabwärts anwachsender Zahlen. Die ersten beiden Ziffern repräsentieren das Haupteinzugsgebiet, die verbleibenden sechs Stellen steigen flussabwärts an. Die Codierung selbst unterscheidet jedoch nicht zwischen Haupt- und Nebenflüssen und stellt keine Informationen zur Struktur des Einzugsgebietsnetzwerkes bereit. Die Zuweisung erfolgt nicht nach hierarchischen Kriterien [WAHL ET AL. 1995].

Die französische Forschungsorganisation ORSTOM (heute INSTITUT FRANÇAIS DE RECHERCHE SCIENTIFIQUE POUR LE DÉVELOPPEMENT EN COOPÉRATION - IRD) entwickelte in den sechziger Jahren ein neunstelliges System für die Zusammenführung gewässerbezogener Daten in Afrika, Südamerika, Europa und Ozeanien. Die erste Ziffer bezeichnet den Kontinent, die zweite und dritte das Land und die vierte und fünfte das kontinentale Einzugsgebiet mit einem Maximum von 99 Haupteinzugsgebieten pro Kontinent. Die sechste und siebente Stelle identifizieren den Fluss und die letzten beiden Ziffern den Abschnitt. Es ist damit gut geeignet, tabellarische Daten nach dem Kontinent, dem Land, dem Einzugsgebiet oder dem Fluss zu ordnen, jedoch ebenfalls wissensbasiert und somit nicht automatisiert abzubilden. Hinzu kommt, dass Veränderungen in Ländernamen oder Territorien, beispielsweise als Resultat politischer Wandlungen, zu Komplikationen führen können [ROCHE 1968].

In Brasilien wurde durch das DEPARTAMENTO NACIONAL DE AGUAS E ENERGIA ELETRICA (DNAEE) ein zehnstelliges Flussordnungssystem eingeführt, das dem des NWIS ähnelt und nur für das Territorium Brasiliens definiert ist [FERNANDES 1987].

Auch das GLOBAL RUNOFF DATA CENTER (GRDC) DER WORLD METEOROLOGICAL ORGANISATION in Koblenz nutzt ein ähnliches, siebenstelliges System, um Tausende von um die Erde verteilten Stationen zur Messung von Abflusswerten räumlich zu klassifizieren [GRDC 1995]. Die erste Ziffer identifiziert den Kontinent, die zweite das Land, die dritte und vierte das kontinentale Einzugsgebiet, die Ziffern fünf, sechs und sieben repräsentieren die Station selbst. Sollten mehr als neun Länder eines Kontinentes einbezogen sein, werden der zweiten Stelle die Werte 1 bis 9 nochmals zugewiesen. Dann hängen allerdings die dritte und vierte Stelle von der zweiten ab. Das System ist somit ebenfalls wissensbasiert.

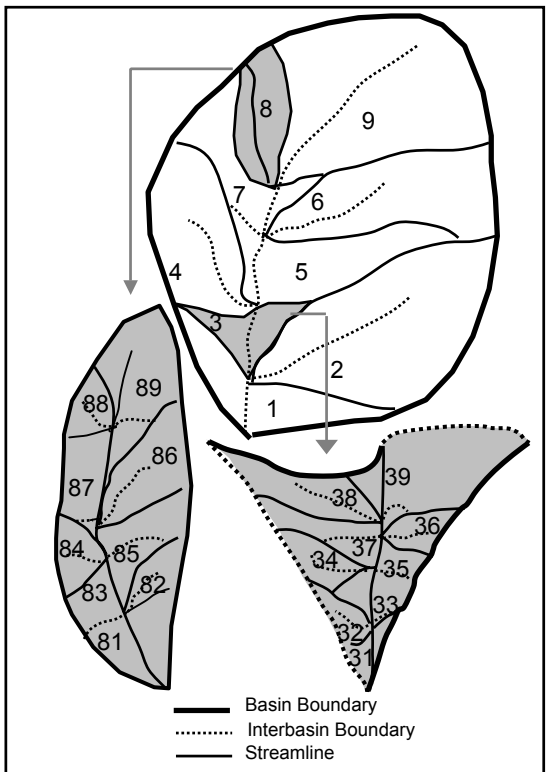


Abb. 7: Einzugsgebietsgliederung und Kodierung (nach [PFAFSTETTER aus VERDIN 1999])

O. PFAFSTETTER, Ingenieur des DEPARTAMENTO NACIONAL DE OBRAS DE SANEAMENTO (DNOS) in Brasilien, schlug 1989 ein natürliches, auf topographischer Kontrolle basierendes System vor (**Abb. 7**), das dem aus dem Oberflächenabfluss der Einzugsgebietsgliederung resultierenden Netzwerk besser als ei-

nige der zuvor beschriebenen Ansätze entspricht und diese natürliche Ordnung in einer zehnstelligen Zahl widerspiegelt: die aufsteigende Ordnung von eins bis neun und die binären Merkmale gerade und ungerade. Die Größe der Ziffer charakterisiert die relative Position zum Gebietsauslass. Ist sie gerade, so liegt der Abschnitt auf dem Hauptfluss, anderenfalls nicht [VERDIN & VERDIN 2000]. Das System ist, verglichen mit den zuvor beschriebenen, effektiv bezüglich der Ausnutzung seiner Ziffern und damit des Speicherplatzes. Es ist in jüngerer Zeit eines der meist beachtetsten Modelle zur hierarchischen Einzugsgebietsgliederung. Da es jedoch primär auf der Einzugsgebietsgliederung basiert und nicht aus der reinen, zweidimensionalen Geometrie der Fließgewässerabschnitte ableitbar ist, kann es in dieser Arbeit nicht verwendet werden.

Ein interessantes Flussordnungsprinzip wurde 1998 von B. LEHNER [LEHNER 1998] vorgestellt. Sein Ziel entsprach in Teilen dem dieser Arbeit. Auch dort sollten aus der zweidimensionalen Geometrie des Gewässernetzes die hierarchischen Zusammenhänge abgeleitet und in einer Tabelle gespeichert werden sowie durch SQL-Abfragen nutzbar sein. Grundlage dieses Ansatzes ist die klassische Flussordnung (**Abb. 6**). In einer Kombination der Ordnung mit Entfernungsangaben wird dieser Anspruch erfüllt. Das System ist jedoch redundant, da die Zahl der benötigten Felder „n.ORD“ mit der Anzahl der Ordnungen zunimmt (**Tab. 1)**.

Node	arc-id	Length	order	1.ord	2.ord	3.ord	4.ord	5.ord
1	645	0	1	0	0	0	0	0
2	786	2436	1	2436	0	0	0	0
3	87	1612	2	2436	1612	0	0	0
4	490	798	2	2436	2410	0	0	0
5	659	853	4	2436	2410	0	853	0
6	141	1272	2	2436	3682	0	0	0
7	99	1878	1	4314	0	0	0	0
8	4711	704	3	4314	0	704	0	0

Tab. 1: Beispieldaten des Systems nach LEHNER [LEHNER 1998]

Die hierarchiebezogenen Querys (Abfragen) erfordern umfangreiche Stringoperationen. Um z. B. vom Gebietsauslass (Node 1) zu Node 8 (Gewässerabschnitt dritter Ordnung) zu gelangen, folgt man der ersten Ordnung 4314 m flussaufwärts und der zweiten Ordnung nach der ersten Verzweigung für 707 m. Im Vorfeld der Selektion müsste der Abfragestring also erst aus vielen Zeilen und Spalten kombiniert werden. Dieses Prinzip lässt sich jedoch auch auf andere Ordnungen, wie denen nach STRAHLER oder SHREVE, aufbauen.

Eine weitere Möglichkeit besteht in der Abbildung von Unter-Oberlieger-Beziehungen (Modell angrenzender Listen, vgl. **Kap. 4.2.2**), für die zwei Felder benötigt werden: eine für die ID jedes Abschnittes und eine andere zur Identifizierung seines flussabwärtigen Nachbarn. Dieses System wurde in der Arbeit untersucht und festgestellt, dass es für die Aufgabenstellung nur eingeschränkt nutzbar, jedoch sehr speichereffizient und intuitiv ist. Darüber hinaus stellt es international in vielen Institutionen einen Quasi-Standard dar, beispielsweise bei hydrologischen Modellierungen [JENSON & DOMINGUE 1988].

Neben diesen Kodierungsmethoden existieren zweifellos andere Schemata, da weltweit viele Organisationen mit Verantwortung im Bereich der Archivierung und des Monitorings hydrologischer Daten tätig sind. Aus der verfügbaren Literatur war jedoch kein System zu erkennen, das den Anforderungen der Aufgabenstellung dieser Arbeit in ausreichendem Umfang gerecht wird. Deshalb wurde über alternative Methoden der Speicherung hierarchischer Attribute in einer für diese Ziele möglichst optimalen Eigenschaftskombination nachgedacht und verschiedene, in späteren Kapiteln ausführlicher beschriebene Ansätze entwickelt und implementiert. Diese Kriterien resultieren aus den in den vorangegangenen Abschnitten beschriebenen Grundlagen und Überlegungen und sind in **Tab. 2** zusammengestellt.

Kriterium
automatisiert aus der 2-dimensionalen Geometrie extrahierbar (also z. B. nicht wissensbasiert)
Eignung für aufwärtsgerichtete Abfragen*
Eignung für abwärtsgerichtete Abfragen*
Eignung für verbindende Abfragen*
iterationsoptimiert bei SQL-freier Selektion[5]
effektiv in der Speicherung bzw. möglichst wenig redundant
erweiterbar / verfeinerbar
regionalisierbar / globalisierbar - es sollte zumindest als Grundlage einer globalen Ausweitung dienen können
intuitiv, d.h. für den Bearbeiter leicht verständlich
hydrologisch aussagekräftig[6]
zur alternativen Darstellung geeignet[7]

Tab. 2: Kriterien der Eignung eines Fließgewässer-Ordnungssystems zur Abbildung in DBMS und GIS

2.4 Datenbanken

Die folgenden Abschnitte stellen das relationale, das objektorientierte und das objektrelationale Datenbankmodell, die Datenbank-Abfragesprache SQL sowie das Konzept Räumlicher Datenbanken vor. Das hierarchische Modell sowie das Netzwerkmodell werden nicht besprochen, da beide heute zunehmend an Bedeutung verlieren [LANG & LOCKEMANN 1995], [JENNINGS 2001].

* mit Standard-SQL
[5] Hier sollen möglichst wenig Durchläufe über eine Tabelle benötigt werden (idealer weise nur einer). Es besteht ein direkter Zusammenhang zwischen SQL- und Schleifenfreundlichkeit: wenn ein Sachverhalt in einer einfachen SQL-Abfrage darstellbar ist, so kann er auch mit einer Iteration abgebildet werden (vgl. **Kap. 4.2 ff.**).
[6] Insbesondere sind hier jedoch Aussagen zur hierarchischen Stellung, wie die Entfernung von der Quelle oder dem Gebietsauslass gemeint.
[7] Z. B. in einem Treeview oder als tabellenbasierter Gewässerbaum in HTML (vgl. **Kap. 4.3 ff.**).

2.4.1 Das relationale Modell

Dieses Datenbankmodell geht auf E. F. CODD [CODD 1969] zurück und wird derzeit am häufigsten eingesetzt. Auch in GIS hat es sich einen wichtigen Platz erobert und wird nach wie vor in Theorie und Praxis eingehend diskutiert [SINGER 1993], [GABRIEL & RÖHRS 1995]. Kennzeichnend für dieses Modell ist die Aggregierung von Datensätzen aus atomaren Komponenten innerhalb sogenannter Relationen. Im Folgenden soll dieser Sachverhalt etwas formaler dargestellt werden.

Seien $A_1..A_n$ paarweise verschiedene Attributnamen und $ID_1..ID_n$ die jeweils zugehörigen Mengen (Domänen), dann heißt jede Teilmenge R aus dem kartesischen Produkt aus $ID_1..ID_n$ eine n-stellige Relation über $ID_1...ID_n$ (**Abb. 8**). Ein Element einer Relation heißt Tupel [HEUER 1997]. Die Domänen selbst bestehen aus atomaren Datentypen wie z. B. Byte, Integer, Float, Double, Boolean, String usw. Diese Datentypen weichen jedoch in den verschiedenen Datenbanksystemen voneinander ab. Es besteht dabei aufgrund des Bildungsgesetzes keine Ordnung innerhalb der Menge der n-Tupel [LANGRAN 1993].

Schlüssel	Fremdschlüssel	NAME
1	32	Fischbach
2	32	Goldbach
3	76	Sandgraben

Schlüssel	Nummer
31	12_32241
32	12_32212
33	12_32256
34	12_32284

Abb. 8: 1/n-Relation zwischen Teileinzugsgebieten (unten) und Flüssen (oben)

Auf dieser Basis sind zahlreiche Operationen denkbar, die ihren Ursprung in der relationalen Algebra haben. Es handelt sich dabei im wesentlichen um ein Abfragemodell, das auf Mengenoperationen wie Vereinigung, Differenz, Projektion und Selektion basiert. Die Schlüsselbedingung besagt, dass zu jedem beliebigen Zeitpunkt t jedes Tupel einer Relation R über ein oder mehrere Attribute A_i eindeutig zu identifizieren ist. Das Attribut, das die Identifizierung des Tupel ermöglicht, nennt man den Schlüssel. Wird die Eindeutigkeit erst über mehrere Attribute erreicht, so spricht man von einem zusammengesetzten Schlüssel [HEUER 1997]. Will man also z. B. alle in einer Tabelle FLUESSE gespeicherten Abschnitte identifizieren, die innerhalb einer Menge M Teileinzugsgebiete liegen, die wiederum in einer anderen Tabelle gespeichert sind, so kann dies innerhalb einer Relation R über einen entweder einfachen oder zusammengesetzten Schlüssel, der in beiden Tabellen angelegt ist, realisiert werden (z. B. die Teileinzugsgebiets-ID in **Abb. 8**). Die Menge aller Tupel lässt sich in Form von Tabellen abbilden. Nicht zuletzt wegen dieser eingängigen Darstellung in Verbindung mit der daraus resultierenden leichten Erlernbarkeit der Abfragesprache SQL hat das relationale Modell eine derartige Verbreitung gefunden.

SQL ist ein Tool für die Kommunikation mit einer relationalen Datenbank zur Definition, Abfrage und Änderung von Daten in einer Datenbank. Mit der SQL-Syntax können Anweisungen konstruiert werden, die Datensätze anhand bestimmter Kriterien extrahieren [VOSSEN 1994], [JENNINGS 2001]. SQL ist eine äußerst leistungsfähige Sprache. Eine einzige Anweisung kann eine gesamte Tabelle verändern. Es existieren viele Versionen von SQL. Jede wurde für ein bestimmtes DBMS (Datenbank-Management-System) entwickelt, sie sind jedoch alle von einem 1992 veröffentlichten und seitdem mehrmals erweiterten ANSI-SQL-Standard abgeleitet, an dem sich diese Arbeit orientiert.

Eine SQL-Abfrage ist eine formalisierte Anweisung an eine Datenbank, entweder eine Datensatzgruppe zurückzugeben oder eine bestimmte Aktion mit einer Datensatzgruppe durchzuführen und beginnt mit einem "Verb"-Schlüsselwort, zum Beispiel `CREATE` oder `SELECT` [VOSSEN & WITT 1988]. Zum Beispiel könnte die folgende SQL-Anweisung die Namen aller Flüsse in Frankreich zurückgeben (sofern diese in einer Tabelle „FLUESSE“ abgelegt sind...):

```
SELECT [RiverName] FROM Fluesse WHERE State = "FRA"
```

RDBMS sind durch Persistenz (dauerhaftes Speichern von Datenobjekten), Unterstützung des effizienten Zugriffs auf die Daten durch Indizes oder Speicherstrukturen, wie z. B. Cluster, Mehrbenutzer-Betrieb sowie Recovery-Mechanismen für den Fall, dass eine Fehlersituation aufgetreten ist, gekennzeichnet.

Mit SQL3 wird gegenwärtig eine stärkere Objektorientierung (vgl. **Kap. 2.4.2**) eingeführt, durch:

- benutzerdefinierte Datentypen (Abstract Data Types, Named Row Types, Distinct Types)
- Typ-Konstruktoren für Zeilen und Referenzen (Pointer)
- Typ-Konstruktoren für Collection-Typen (Sets, Lists, Multi Sets)
- benutzerdefinierte Funktionen und Prozeduren
- Unterstützung grosser Objekte wie BLOBs (Binary Large Objects) und CLOBs (Character Large Object)

SQL3 wird bisher jedoch nur von wenigen Datenbanken unterstützt (POSTGRESQL©, MYSQL 4.1©, ORACLE 9I™).

2.4.2 Das objektorientierte Modell

„Auch wenn der Reifegrad objektorientierter Datenbanken noch gesteigert werden kann, ist dieser zukunftsgerichteten Technologie ein Durchbruch in der Praxis zugesichert ...“ [MAIER & BACHMANN 1997, S.17]. Obwohl es, im Gegensatz zum relationalen Modell, für objektorientierte Datenbanksysteme noch keine verbindliche internationale Norm gibt, soll dieser Ansatz aufgrund seiner zunehmenden Bedeutung hier mit betrachtet werden. Das babylonische Sprachgewirr ist hinsichtlich der Terminologie zum objektorientierten Modell besonders ausgeprägt. D. FRITSCH und K. - H. ANDERS [FRITSCH & ANDERS 1996] versuchten, mit einer feinen Differenzierung der Begriffe strukturell, verhaltensmässig und voll objektorientiert etwas Licht in das Dunkel zu bringen (vgl. auch [LANG & LOCKEMANN 1995], [HEUER 1997] und [JENNINGS 2001]). Speziell im Zusammenhang mit GIS wurde dieses Konzept von R. BILL [BILL 1996] beschrieben. Im Moment befinden sich allerdings noch wenige solcher Systeme im Einsatz (wie z. B. OBJECTSTORE™ von OBJECTDESIGN oder POET™).

Objekte (**Abb. 9**) sind Grundbausteine objektorientierter Anwendungssysteme. Jedes Objekt ist eine eigenständige Einheit, die über Daten (Attributswerte des Objektes) und Verhalten (Methoden oder Operationen des Objektes) verfügt. Objekte mit identischer Struktur und gleichem Verhalten werden zu Klassen abstrahiert. Auf die Daten (Attribute) innerhalb eines Objektes kann nur mithilfe der definierten Methoden zugegriffen werden. Ein direkter Zugriff auf Daten ist somit von außen nicht möglich. Attribute und Methoden werden auch als Komponenten eines Objekttyps bezeichnet.

Objektorientierte Datenbanken (OODBMS) unterscheiden sich konzeptionell stark von relationalen Datenbanken. Es werden z. B. nicht nur Daten (Attribute) der Objekte, sondern auch ihr „Verhalten" (Methoden oder Funktionen) in der Datenbank gespeichert. Beziehungen zwischen verschiedenen Objekten werden mithilfe vom DBMS vergebener, systemweit eindeutiger „Objektidentitäten" (OID) hergestellt. Die Speicherung lässt sich nicht so einfach beschreiben wie die relationaler Datenbanksysteme, da Attribute und Funktionen meist nicht gemeinsam abgelegt werden. OODB-Systeme übernehmen die sehr positiven Eigenschaften ihrer Vorgänger und verfügen außerdem über folgende Merkmale:

- die üblichen, vom System vordefinierten Standarddatentypen, wie z. B. Integer-, String-, oder Character-Datentypen
- die typischen Konzepte der Objektorientierung: Klassen und Methoden, d.h. Operationen auf den Objekten der Klasse
- Konzepte zur Beschreibung komplexer Objekte.

Dazu werden Typkonstruktoren in die Welt der Datenbanken eingeführt. Es handelt sich dabei um Sprachelemente, mit denen komplexe Datentypen (Klassen) aufgebaut werden können, wie in **Abb. 9** und in **Code 1** dargestellt.

CODE 1: SCHEMA EINER IMAGINÄREN KLASSE INNERHALB EINES OODBMS

```
Class Abschnitt{
 Properties
  Dimension: Line;
  Name: String;
  FGWID, Unterlieger, Gebietsauslass:  Integer;
  Laenge, Q, Mng: Double;
  istQuelle: Boolean;
 Operations
  sucheNachbarn(FGWID): Datenstruktur;
  findeAlleOberlieger(FGWID): Datenstruktur;
  wegZumAuslass(FGWID, Gebietsauslass): Datenstruktur;
  sucheVerbindung(FGWID, Integer): Datenstruktur;
}
Class Quelle Inherits Abschnitt{
 Properties
  istQuelle: Set (Nachbarn) inverse is Nachbarn.count = 1;
 Operations
  gibHierarchischeEntfernungAn (Gebietsauslass);
  gibHierarchischeBreiteAn (Gebietsauslass);
}
```

Die Eigenschaften und Verhaltensweisen eines Abschnittes – eines Objektes der Klasse „Abschnitt“ – könnten auf diese Art schon im OODBMS so angelegt werden, dass sich hierarchische Querys direkt an den Abschnitt stellen lassen. Es sind jedoch noch eine Reihe von Problemen heutiger OODBMS mit verschiedenen Ursachen sichtbar [HEUER 1997]:

- Es gibt unterschiedliche Konzepte verschiedener Anbieter, unter denen sich noch kein Markt- bzw. Technologieführer herausgebildet hat.

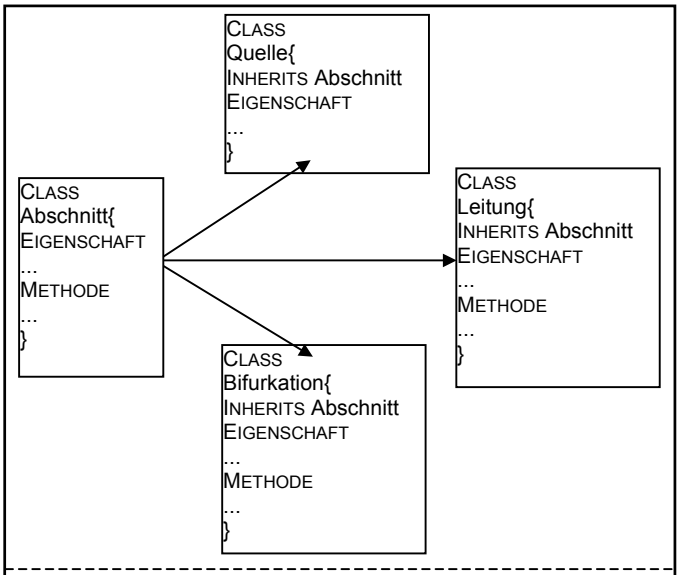


Abb. 9: Spezialisierung in einer Klassenhierarchie (nach [HEUER 1997])

- Die Erstellungs- und Zugriffssprachen sind nicht standardisiert.
- Sie sind langsamer als relationale Systeme. Aufgrund der komplexen Strukturen sind sowohl die Speicherung von z.T. vielfach verknüpften Objekten als auch der Zugriff (Wie weit soll sich der Zugriff im „Objektnetz“ erstrecken?) aufwendiger, als bei relationalen Systemen [GEPPERT 1999].
- Nur wenige kommerzielle Systeme sind verfügbar und noch weniger produktive Einsätze bekannt. Der Anteil kommerziell eingesetzter OODB liegt im Bereich weniger Prozente. Der Markt für kommerzielle Datenbanken, die das Prädikat "objektorientiert" für sich in Anspruch nehmen, ist stark in Bewegung (z. B. GEMSTONE™ der Fa. SERVIO LOGIC, auf Smalltalk-80™ basiert, O2™ von der Firma O2 TECHNOLOGY mit eigener objektorientierter Sprache O2C sowie Schnittstellen zu C, C++, JAVA und SMALLTALK, seit 1990 OBJECTSTORE™ von OBJECTDESIGN mit C++-ähnlicher Abfragesprache).

Aufgrund dieser Konstellation von Vor- und Nachteilen wurden die Möglichkeiten zur Speicherung von Fließgewässerhierarchien in OODBs hier nicht weitergehend untersucht. Es ist jedoch damit zu rechnen, dass sich diese Technologie über einen längeren Zeitraum hinweg durchsetzen wird, da die Anforderungen, die Entwurf und Verwaltung komplexer Objekte mit sich bringen, einen Evolutionsschritt in der Datenbanktechnologie förmlich erzwingen [DARWIN 1859], [GEPPERT 1999].

Auf diesem Weg ist auch das objektrelationale Modell, bei dem bestehende relationale Systeme um objektorientierte Eigenschaften erweitert werden. Dabei wird insbesondere darauf geachtet, den Sprachstandard für SQL zwar auszudehnen, aber zur bisherigen Normung kompatibel zu bleiben [VOSSEN 1994]. Diese Spracherweiterungen orientieren sich weitgehend am vorgeschlagenen Standard SQL3. Dieser hat das Ziel, ein standardisiertes objektrelationales Modell einschließlich einer standardisierten Abfragesprache zu entwickeln. Viele Hersteller relationaler Datenbanken arbeiten an einer objektorientierten Erweiterung des Datenmodells (z. B. ORACLE™, PostGreSQL©). Deshalb werden objektorientierte Datenbanken vor allem von den „üblichen“ Herstellern relationaler Systeme angeboten. Im Allgemeinen haben objektrelationale Datenbanken folgende Zusatzeigenschaften (verglichen mit den RDBMS, bei verschiedenen Anbietern unterschiedlich ausgeprägt und realisiert) [HEUER 1997]:

- ein erweitertes, benutzerdefiniertes System neuer Datentypen und Zugriffsmethoden
- das Speichern von Operationen
- komplexe Objekte
- Vererbung
- Polymorphismus.

Hierarchiebasierte SQL-Statements würden demnach auch in einer solchen Umgebung möglich sein, sofern sie sich am ANSI-SQL-Standard orientieren.

2.4.3 Räumliche Datenbanken (RDB)

Um den Funktionsumfang der Ergebnisse der Arbeit auch auf die zunehmend Verbreitung findenden (wenn auch meist noch sehr kostenintensiven) räumlichen Datenbanken auszuweiten, wurde untersucht, inwieweit die zugrunde liegenden Prämissen der Übertragbarkeit und Verfügbarkeit diese Technologie einschließen können. In den vergangenen Jahrzehnten ist die Zahl räumlicher Datenbanken die GIS unterstützen, stark angestiegen [GUTING 2000]. Unter der Vielzahl auf dem Markt befindlicher Produkte seien exemplarisch die folgenden, wohl meistverbreiteten Systeme genannt: IBM’S DB2 UNIVERSAL DATABASE™, INFORMIX DYNAMIC SERVER SPATIAL DATABLADE™, ORACLE SERVER™ mit der

SPATIAL DATA OPTION™, G.SERVER™ von GEOTASK sowie POSTGIS©. Diese letztgenannte RDB ist im Internet frei verfügbar und daher besonders interessant. Sie basiert auf dem, ebenfalls als Opensource verfügbaren, sehr leistungsstarken objekt-relationalen Datenbanksystem POSTGRESQL©, das darüber hinaus sehr gut mit PHP (Hypertext Preprocessor, vgl. **Kap. 4.3.2**) und dem MAPSERVER© (Opensource) der UNIVERSITY OF MINNESOTA harmoniert. Die Funktionalität von PostGIS entspricht der teurer Produkte wie ESRIs ArcSDE™ oder der ORACLE SPATIAL DATA ENGINE™. Leider erfordert das System unter Windows™ die CYGWIN UNIX EMULATION LAYER©, deren Installation und korrektes Handling nicht generell ohne Probleme sind. Damit kam es für diese Arbeit nicht in Frage.

RDB kombinieren Know-How aus drei unterschiedlichen Wissenschaftsbereichen: Datenbanken und Informationssysteme, Geographie und Kartographie sowie abstrakte und Computer-Geometrie und Topologie [KEMPER & WALLRATH 1987]. Sie vereinen GIS mit klassischen SQL-Datenbanken und ermöglichen ein besseres Datenmanagement im räumlichen Bereich durch folgende Eigenschaften (**Abb. 10**):

Art	Dichte	Vorkommen
Esche	20	
Buche	10	
Eiche	80	

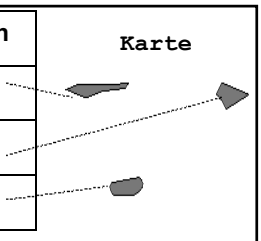


Abb. 10: Schema der räumlichen Erweiterung innerhalb eines RDB (nach [LAURINI & THOMPSON 1992])

- Datenintegrität und redundanzfreie Verwaltung großer Datenmengen
- schneller Zugriff und optimierte Abfragemöglichkeiten
- einheitliche Umgebung zur Verwaltung räumlicher Daten und der Standarddatenbank.

Das RDB-Datenmodell neueren Typus nutzt den objektorientierten Ansatz und unterstützt Punkte, Linien und Polygonobjekte. Linienobjekte sind dabei Arrays von Punkten und Polygone Linien, deren Anfangs- und Endpunkt identisch sind. Sich selbst kreuzende Linien basieren auf sogenannten Spaghetti-Strukturen. Polygone mit Inselobjekten werden als Doughnut-Polygone bezeichnet. Das Datenmodell speichert die räumliche Repräsentation eines Objektes in einer einzigen Zeile, unabhängig von seinem Typ. Informationen zur Flächengröße, Umfang (Länge) oder die Umfassungskoordinaten sind mit in diesem räumlichen Bereich gespeichert. Jedes Objekt hat einen räumlichen Index zur Effektivierung räumlicher Abfragen. Die räumlichen, attributiven und Indextabellen bilden eine GIS-Informationsschicht – oft als Layer oder Thema bezeichnet. Dabei ist es nicht erforderlich, dass die Objekte innerhalb eines Layers vom selben Typ sind [GUTING 2000].

Die Überprüfung der Ansätze der Arbeit im Umfeld einer RDB wurde freundlicherweise vom UFZ Leipzig-Halle mit der dort vorhandenen Soft- und Hardwarekombination von ORACLE SERVER mit der SPATIAL DATA OPTION und der SPATIAL DATABASE ENGINE (SDE) von ESRI ermöglicht. Bei SDE handelt es sich um einen transaktionsorientierten Anwendungs-Server, der ORACLE als Datenbankbasis nutzt (**Abb. 11**). Die SDE-Technik erlaubt eine äußerst schnelle Selektion von Daten aus großen geographischen Informations-Pools. Die von ORACLE und ESRI veröffentlichten Schnittstellen (APIs) zu SDE sollen gewährleisten, dass sich die Visualisierungs- und Analysefunktionen auch innerhalb bestehender Client-Server-Anwendungen (z. B. verbunden mit ARC/INFO, ARCVIEW, ARCGIS, MAPOBJECTS oder ARCIMS™) realisieren lassen [ESRI 2001].

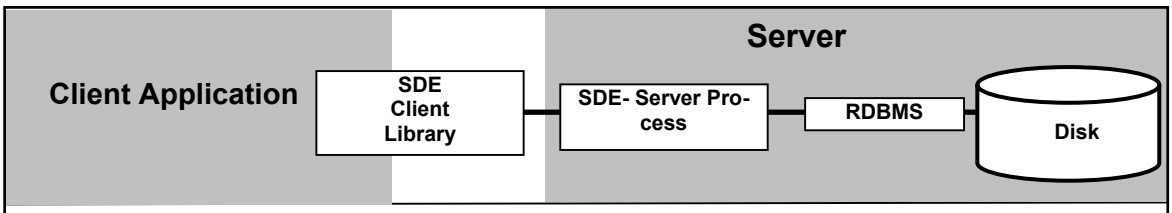


Abb. 11: SDE-Architektur (nach [ESRI (a) 1999])

Auf der Serverseite stehen der SDE-Server-Prozess, das RDBMS, in diesem Fall ORACLE, sowie die aktuellen Daten. Der Server führt in der Regel alle räumlichen Abfragen durch und sendet nur diejenigen Daten an den Klienten zurück, welche den räumlichen bzw. attributiven Abfragekriterien entsprechen. Diese Vorgehensweise ist bei großen Datenmengen (abhängig von der Leistungsfähigkeit der Rechner und des Netzwerkes) wesentlich effektiver, als alle Daten und Prozesse über das Netzwerk auf die Client-Seite zu transferieren [ESRI 2001].

Aus der allgemeinen Charakterisierung räumlicher Datenbanken ergibt sich also die Möglichkeit, SQL-basierte Abfragen auch auf räumliche Objekte in der Datenbank anzuwenden. Im Fall von SDE sind dies sogenannte „FilterExpressions“. Sie sind als verkürzte SQL-Statements aufzufassen und somit in gleicher Weise wie die attributiven GIS-Abfragen durch einfache Stringoperationen aus vorhandenen SQL-Statements abzuleiten, wie das folgende Beispiel zeigt:

```
SELECT * FROM Fluesse WHERE  State = "FRA" //State = "FRA" ist hier die FilterExpression
```

Es ist also nur erforderlich, die ohnehin auf der Attributseite notwendigen Abfragenformulierungen innerhalb der Benutzerschnittstelle auf das Format eines solche Filterausdrucks zu reduzieren. Das Verfahren muss deshalb in den weiteren Ausführungen nicht näher beleuchtet werden.

2.5 Geographische Informationssysteme (GIS)

2.5.1 Einführung zu Geographischen Informationssystemen

Die Zahl verschiedener Definitionen für Geographische Informationssysteme ist nahezu unübersehbar. Eine Ursache für diese Vielfalt liegt sicher darin, dass sich GIS mit komplexen Themen aus den unterschiedlichsten Disziplinen beschäftigen. Eine einfache und kompakte Definition stammt aus dem englischen Sprachraum: *"A Geographic Information-System (GIS) can be defined as a system containing a spatial database representing aspects of a cultural and physical environment of a particular geographic region together with procedures for analysing combinations of attributes and generating graphical or statical products..."* [WILKINSON ET AL. 1986 in: SAURER & BEHR 1997 S.6]. Das zentrale Kriterium eines GIS ist die enge Integration geometrischer und thematischer Attribute von räumlichen Objekten. Während Geometriedaten die Lage eines Objektes in der Umwelt beschreiben, beinhalten Sachdaten die Eigenschaften (Attribute) dieses Objektes [BILL & FRITSCH 1991].

Mit einem GIS können große Datenmengen verwaltet und schnell nutzbar gemacht werden. So ist es z. B. möglich, die vielen in unterschiedlichen Formaten und Maßstäben in den öffentlichen Verwaltungen vorliegenden Daten zur Analyse und Weiterverarbeitung

zusammenzuführen. Auf diese Art können mithilfe geeigneter Funktionen aus der Kombination der Ausgangsdaten neue Informationen gewonnen werden.

Grundsätzlich wird zwischen Raster- und Vektordaten unterschieden. Vektordaten können mit einer Vielzahl unterschiedlicher Attribute (z. B. Flächennutzung) verknüpft werden. Rasterdaten besitzen, abgesehen von ihrem Grau- bzw. Farbwert, keine Attribute und sind im analytischen Bereich (z. B. bei Ausbreitungs- und Distanzanalysen) von großer Bedeutung [BILL & FRITSCH 1991]. Hybride Geo-Informationssysteme sind in der Lage, mit Vektor- und Rasterdaten zu arbeiten und z. B. Höhenmodelle und räumliche Statistiken in Form von TINs (Triangulated Irregular Network) oder Grids abzuleiten[8] [BARTHELME 1995], [BÜYÜKSALIH & JACOBSEN 2002].

Moderne GIS-Systeme fassen Geometrie- (Punkte, Linien, Flächen) und Sachdaten als Einheiten bzw. Objekte auf. Diese Objekte mit ihren Eigenschaften definieren das Verhalten und die Möglichkeiten eines sogenannten objektorientierten GIS (vgl. auch **Kap. 2.4.2**). Die Objektorientierung betrifft vor allem die Modellbildung innerhalb des Systems [LIEBIG 1997]. GIS-Objekte sind z. B. Fließgewässerabschnitte, Teileinzugsgebiete, Messstellen, aber auch Grafiken, Layouts oder Tabellen. Die Klassen sind durch eine hierarchische Struktur gegliedert (**Abb. 12**) und besitzen Eigenschaften und Methoden, mit deren Hilfe die von diesen Klassen abgeleiteten Objekte beeinflusst werden können. Objekte verschiedener Klassen sind damit auch in der Lage, miteinander zu kommunizieren [MEYERS 1992].

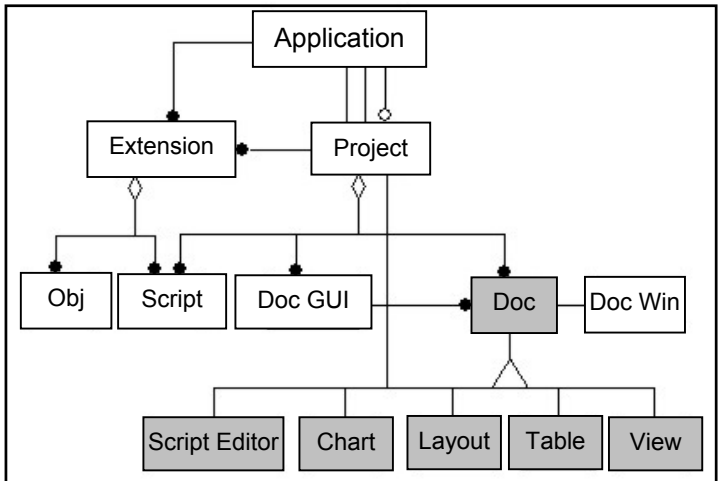


Abb. 12: Ausschnitt aus der Klassenhierarchie von ArcView ("Application Framework") [ARCVIEW-Online-Hilfe]

Im Gegensatz zu prozeduralen Sprachen wie C oder PASCAL, deren Programmcodes eine Liste von Anweisungen darstellen, ist die Struktur einer objektorientierten Programmiersprache zunächst vor allem darauf ausgerichtet, Prozesse und Algorithmen in computergerechter Weise zu formulieren und dadurch ergonomische Instrumente zur Nachbildung realer Prozesse zu schaffen. Ihre Anlehnung an die menschliche Vorstellung (Objekte - Klassen - Eigenschaften - Vererbung von Eigenschaften usw.) ermöglicht die Bewältigung komplexer Probleme in relativ nachvollziehbarer Form. Vier Hauptmerkmale kennzeichnen die objektorientierte Programmierung [BOOCH 1994]:

- Aktionen werden durch Nachrichten erzeugt.

[8] An dieser Stelle sei noch einmal darauf hingewiesen, dass es in dieser Arbeit nicht um die Ableitung der Fließgewässerstruktur aus Höhenmodellen geht. Das ist in modernen GIS-Systemen in der Regel - zumindest technisch - unproblematisch und kann, sofern diese Befehle nicht schon in der Benutzerschnittstelle integriert sind, meist mit wenigen Zeilen Code realisiert werden. Die Ableitung der Gewässerstruktur und ihrer Ordnungen aus digitalen Geländemodellen (DGM) beschreiben unter anderem [VAN DEURSEN & KWADJIK 1990], [HEUVELINK 1993] und [BLASCHKE 1997]. Inhaltlich ergeben sich dabei Fehlerquellen, auf die hier jedoch nicht näher eingegangen werden kann (vgl. dazu z. B. [JENSON & DOMINGUE 1988], [KUMLER 1994] und [BARRINGER & LILBURNE 1997]).

- Die Zugehörigkeit des angesprochenen Objektes (Instanz) zu einer Klasse bestimmt die Art der Aktion, die ausgeführt wird.
- Vererbung: Klassen sind hierarchisch strukturiert und können somit ihre Eigenschaften (Attributdefinitionen) und Methoden vererben (**Abb. 21**).
- Nachrichten können durch diese Hierarchie hindurch weitergeleitet werden.

Dem GIS-Anwender wird so die Möglichkeit gegeben, seine Datenmodelle frei zu definieren und spezielle Eigenschaften hinzuzufügen, die im Standardsystem nicht enthalten sind [LIEBIG 1997].

ARCVIEW folgt mit der Programmiersprache AVENUE dem objektorientierten Ansatz, *"...obwohl ArcView noch nicht als Object-GIS im strengen Sinne anzusehen ist ..."* [LIEBIG 1997, S. 12]. So können z. B. keine eigenen Klassen entwickelt werden (vgl. **Kap. 2.4.2**).

In der Welt der Vektoren wird jegliche Geometrie durch den Punkt ausgedrückt [BLASCHKE 1997]. Im zweidimensionalen Raum wird beispielsweise die Gerade durch zwei Punkte und die Fläche durch einen geschlossenen Polygonzug dargestellt. Auf Basis dieser Vorgaben sind zahlreiche Datenstrukturen denkbar. Dabei kann im wesentlichen zwischen geometrischen und topologischen Vektordatenmodellen unterschieden werden [FINDEISEN 1990]. Beide Gruppen kennen eine Vielzahl von Implementierungen, wobei letzterer jedoch im Bereich GIS die größere Bedeutung zukommt [RÖSCH 1998]. Die Topologie eines geometrischen Objektes ist durch seinen Anfangs- und seinen Endpunkt (From- und To-Node) sowie seinen linken und rechten Nachbarn definiert und beschreibt die räumlichen Beziehungen geographischer Elemente wie Punkte, Linien und Polygone zu anderen. Beispiele solcher topologischen Fragestellungen wären:

- Welche Arten von Landnutzung grenzen an ein Gewässer?
- Was ist der kürzeste Weg von Punkt A nach Punkt B?

ARC/INFO beispielsweise nutzt ein topologisches Datenmodell in Form von Coverages. Es wäre sehr kompliziert und zeitaufwendig, diese Informationen ohne Topologie zu erhalten.

2.5.2 ARCVIEW als Entwicklungsumgebung einer GIS-Applikation

Der Prämisse dieser Arbeit folgend, die Implementierung der Ergebnisse innerhalb möglichst weit verbreiteter und leicht zugänglicher Softwareprodukte zu realisieren, wurde als Framework der Erstellung eines Moduls zum hierarchieorientierten Präprozessing das sowohl in Deutschland als auch international am weitesten verbreitete Desktop-GIS ARCVIEW gewählt [BUHMANN & WIESEL 1998].

War ARCVIEW anfangs hauptsächlich ein vektorbasiertes System zur Kartenerstellung, führten Weiterentwicklungen zu einem umfassenden, hybriden GIS, das nahezu alle Aspekte im Bereich der geographischen Datenverarbeitung abdeckt. So existieren mittlerweile Module für die Rasterdatenverarbeitung inklusive Fernerkundungsanwendungen, die Modellierung von Oberflächen, Funktionen zur Integration externer Datenbanken und Bildformate sowie viele Fremdentwicklungen, die diese GIS-Umgebung zu einem der beliebtesten Werkzeuge im Bereich der Geoinformation machen. ARCVIEW ist multiplattformtauglich[9]. Ein ARCVIEW-Projekt mit all den möglicherweise bisher bereitgestellten Funktionalitäten in AVENUE würde sowohl unter 16- und 32-bit WINDOWS, verschieden U-

[9] Auf Grund der „Open Interface"-Elemente von NEURON DATA®.

NIX-Derivaten und auf MACINTOSH in nahezu derselben Form erscheinen [GANTER 2001][10].

Spezielle Applikationen können mithilfe der integrierten Scriptsprache AVENUE, die Variablen, Schleifen, Daten- und Entscheidungsstrukturen etc. beinhaltet, erstellt werden. Darüber hinaus bietet ARCVIEW in verschiedenen Bereichen Programmierschnittstellen, wie z. B. die C-Schnittstelle der 3D-Analyst-Extension an.

ARCVIEW kann Shapefiles, CAD-Files, SDE-Sources und Coverages darstellen. Das Standard-Geometriedatenformat sind jedoch Shapefiles. Ein Shapefile speichert Geometrie- und Attributinformationen räumlicher Objekte in Form von Punkten (auch Multipoint), Linien und Polygonen nicht-topologisch. Attribute werden in einem dBASE©-File gespeichert. Jeder Datensatz hat dabei eine „one-to-one"- Beziehung mit dem korrespondierenden Datensatz im Shape. Die Geometrie ist in den Shapes als Datenstruktur von Vektorkoordinaten abgelegt. Die Vorteile liegen in der größeren Darstellungs- und Bearbeitungsgeschwindigkeit gegenüber topologischen Datenstrukturen. Typischerweise benötigen sie auch weniger Speicherplatz und sind einfacher zu lesen und zu schreiben [ESRI 1998]. Andererseits bringt dieses Speicherformat auch eine Reihe von Nachteilen mit sich. So erlauben Shapefiles z. B. Überlagerungen, Kreuzungen, Lücken und nicht geschlossene Polygone. Diese sind im topologischen Sinne nicht „sauber" und müssen vor einer Hierarchisierung (vgl. **Kap. 4.2**) bereinigt werden.

Unter ARCVIEW wird eine temporäre Topologie erzeugt (vgl. **Kap. 2.5.1**). Bei Fehlern in dieser Topologie ist es jedoch vergleichsweise kompliziert, wenn auch nicht unmöglich, ihre Ursache zu finden und diese Probleme zu beheben [ESRI 2000]. In den folgenden Ausführungen wird darauf noch näher einzugehen sein.

2.5.3 MAPOBJECTS als Instrument zur Verifizierung der Ansätze

Zur Überprüfung der Ansätze wurden einige Funktionalitäten auch in VISUAL BASIC 6 mit MAPOBJECTS umgesetzt. MAPOBJECTS ist eine GIS-bezogene Softwarekomponte von Programmiersprachen wie VISUAL BASIC, DELPHI™ oder C++. Verglichen mit ARCVIEW gestaltet sich der Funktionsumfang dabei recht bescheiden. Es ist jedoch möglich, wenn auch meist mit wesentlich größerem Aufwand, die in AVENUE oft sehr mächtigen Befehle unter dieser Entwicklungsumgebung nachzubilden. Allerdings sind diese Funktionen nur unter 32-bit-WINDOWS lauffähig. Innerhalb solcher Restriktionen kann MAPOBJECTS jedoch sehr gute Dienste leisten. Durch die Einbindung der GIS-Komponente in den bereits vorhandenen Sprachumfang ergibt sich eine weitaus größere Flexibilität in sogenannten "non-mapcentric" Applikationen, die hervorragend mit anderen Sprachkomponenten zusammenarbeiten können [GANTER 2001]. Hier ist diese Integration wichtiger, als die reine GIS-Anwendung. Wo ARCVIEW ein GIS-Framework ist, kann MAPOBJECTS mit einer Menge anderer Elemente wie zum Beispiel Video-Player, Voice-Synthesizer, Analog-Digital-Konvertern, Fax, Telefon, u.s.w. zusammenarbeiten. Wie MAPOBJECTS werden diese Komponenten in die Entwicklungsumgebung integriert und untereinander mit Programm-Code verbunden.

MAPOBJECTS ist eine OLE 2.0 bzw. OCX (Object Linking and Embedding) -Control. Darüber hinaus beinhaltet es etwa 45 weitere Objekte, die mittels OLE-Automation bereitgestellt werden. OLE ermöglicht eine Verbindung (Linking) und Einbettung (Embedding) von

[10] Dies trifft z. B. für seinen Nachfolger ArcGIS 8.x nicht zu.

Dokumenten wie z. B. einer Karte innerhalb eines WORD-Dokumentes. Server wie MAPOBJECTS und Klienten bzw. Container wie VISUAL BASIC, EXCEL™ oder WORD™ (viele OLE-Container können gleichzeitig OLE-Server sein) kommunizieren auf diese Art. OLE wiederum baut auf der COM- (Component Object Model)Technologie von MICROSOFT auf. COM und OLE sind integrale Bestandteile von 32-bit WINDOWS wie WINDOWS 9x, WINDOWS NT, 2000 und XP.

MAPOBJECTS ist also eine Software-Komponente mit im GIS-Bereich weitaus weniger Infrastruktur als ARCVIEW, doch zugleich wesentlich flexibler als dieses und mehr darauf ausgerichtet, mit anderen Komponenten zusammen zu arbeiten. Außerdem lassen sich auf Basis der zugrundeliegenden Programmiersprache eigenen Klassen kreieren. Es kam vor allem auch bei der in **Kap. 4.3.1** beschriebenen Anwendung zu einem Einzugsgebiets-Monitoring-System zum Einsatz.

Im Laufe der Arbeit wurde versucht, ausgewählte Bereiche innerhalb der AVENUE-Lösungen, die hinsichtlich der Übertragbarkeit als besonders wichtig anzusehen waren, mit MAPOBJECTS zu verifizieren. Ansatzweise werden die folgenden Kapitel näher darauf eingehen. Die eigentlichen Applikationen zur Datenkorrektur wurden innerhalb von ARCVIEW umgesetzt.

Insgesamt kamen folgende Werkzeuge und Techniken zum Einsatz:

Datenaufbereitung (Präprozessing) und Hierarchisierung	ArcView™-GIS Avenue© (ArcView – Makrosprache) Visual Basic™ C++ MapObjects™
Fließgewässermanagementsystem	SDE™ (Spatial Database Engine) SQL (Structured Query Language) Oracle™ (Client/Server – Datenbank) MsAccess™-Datenbank Visual Basic
Internet-Applikation	SQL (Structured Query Language) MySql© (Opensource-Datenbank) PHP© (Opensource-Programmierumgebung) Javascript© Java© (Geotools) Mapserver© (Opensource-Mapserver)

Tab. 3: In der Arbeit verwendete Programme und Techniken

3 Testgebiete und Datenbasis

Das Einzugsgebiet der SALZA, einem linken Nebenfluss der Saale, liegt im Westen der Industrieregion Halle – Leipzig – Bitterfeld und erreicht mit ungefähr 565 km^2 ca. ein Neuntel der Gesamtfläche des Regierungsbezirkes Halle. Es gehört den vier Landschaftseinheiten Mittel- und Unterharz im Nordwesten, östliches Harzvorland im Nordosten, Helme-Unstrut-Schichtstufenland im Südwesten und Querfurter Platte im Südosten an [LAU 1997], [KRUMBIEGEL & SCHWAB 1974] (**Abb. 13**). Die Leelage zum Harz stellt es in die Reihe der niederschlagsärmsten Gebiete Deutschlands (Mitteldeutsches Trockengebiet). Dabei erreichen die Jahresamplituden der Gebietsniederschläge vor allem mit in den Sommermonaten auftretenden Starkregenereignissen hohe Werte [SCHMIDT 1997]. Diese Besonderheiten von Evapotranspiration und Niederschlag haben auf die Abflussverhältnisse der Oberflächengewässer großen Einfluss. Im Norden findet sich ein zentripetales Oberflächenentwässerungsnetz mit den Mansfelder Seen als Mittelpunkt und im Süden ein baumartig verzweigtes Oberflächenentwässerungsnetz, wobei besonders die tektonischen Störungszonen, wie z. B. der Martinschächter Flözgraben (Böse Sieben) Abflussbahnen in Form von Tiefengräben vorgeben [KNITZSCHKE 1995].

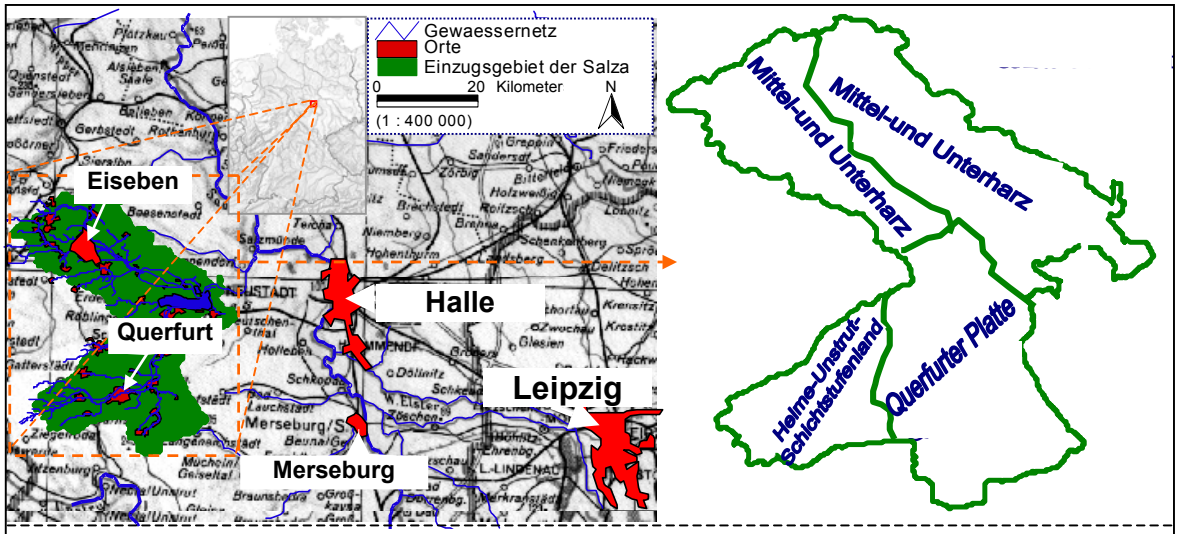


Abb. 13: Lage des Untersuchungsgebietes mit Gewässernetz, Ortschaften und Einzugsgebiet der SALZA
rechts: Landschaftseinheiten im Einzugsgebiet der SALZA

Löß und Lößderivate stellen das bedeutendste Ausgangssubstrat der Bodenbildung im Einzugsgebiet der SALZA dar. Räumliche Differenzen der Witterungsverhältnisse und des Mesoreliefs führten zu unterschiedlichen Bodenformen, von denen Löß-Schwarzerde, Löß-Braunschwarzerde, Löß-Pararendzina und Löß-Parabraunerde die bedeutendsten sind [SCHMIDT 1997]. Nur im Südwesten des Untersuchungsgebietes weichen die Verhältnisse großflächig ab. Hier, im Ziegelrodaer Plateauhügelland, stellt verwitterter Buntsandstein das Ausgangssubstrat vor allem von Berglehm/Bergton- Braunerden und Braunstaugleyen [SCHRÖDER 1986]. Im Bereich der Mansfelder Seen sind Gleye und Schwarzgleye ausgebildet. In den Plateaulagen der Unterharzhochfläche entwickelten sich stauwasserbeeinflusste Pseudogleye [SCHMIDT 1997], [FRIEDRICH & FRÜHAUF 2002].

Aufgrund der äußerst günstigen pedologischen Voraussetzungen begann die landwirtschaftliche Intensivnutzung (heute v.a. Getreide, Hackfrüchte, Mais, Sonnenblumen und Raps) bereits relativ früh. Die pedologische Situation der Mansfelder Mulde ist durch großflächige, zum Teil erhebliche Schwermetallkontamination gekennzeichnet [SCHMIDT

ET AL. 1992], wobei die mittlere bis hohe Erosionsdisposition des Gebietes [SCHRÖDER 1986] aufgrund der flächenhaften Verbreitung von Löß sowie der geringen räumlichen Niederschlagsvariabilität vor allem durch Reliefparameter differenziert wird [SCHMIDT & FRÜHAUF 1997]. Erosion und intensive Nutzung führten zu teilweise erheblichen Störungen der natürlichen Bodenfunktionen. Wälder haben an der Gesamtfläche nur einen geringen Anteil (vorwiegend Traubeneichen-/Eichen-Rotbuchen-/Buchen-Waldgesellschaften, in Tallagen Schwarzerlen – Esche – Wälder / Bruchwälder, Hainbuchenwälder auf südexponierten Muschelkalkhängen).

Die Abflussmenge der den im Westen der Mansfelder Mulde ausstreichenden Zechstein mit ihren Oberläufen querenden Gewässer (Dippelsbach, Kliebigsbach, Viezbach), wird durch Versickerungserscheinungen dezimiert [JANKOWSKI 1995]. Die im gesamten Untersuchungsraum vorherrschenden, zur Oberflächenabflussbildung neigenden schluffigen und tonigen Substrate unterstützen die engen aber hohen Amplituden des Abflussganges.

Die natürlichen Grundwasser-Abflussverhältnisse im Einzugsgebiet der Mansfelder Seen wurden seit Ende des 15. Jahrhunderts durch den in die Mansfelder Mulde vordringenden Kupferschieferbergbau irreversibel verändert. Die Weiterentwicklung der Bergbautechnik ermöglichte immer tiefere Absenkungen des Grundwassers und seine Ableitung in Wasserhaltungsstollen (z. B. Froschmühlenstollen, Zabenstedter Stollen, Schlüsselstollen) [JANKOWSKI 1995], [FRIEDRICH & FRÜHAUF 2002]. Um 1900 häuften sich jedoch die Wassereinbrüche in die Gruben des Südwest-Teils der Mansfelder Mulde, deren Höhepunkt das Ablaufen des Salzigen Sees darstellte. Über die genannten Wasserhaltungsstollen werden den Vorflutern auch heute noch große Mengen an gelösten Salzen (Sulfat, Chlorid) und des Schwermetalls Zink zugeführt [SCHMIDT 1997]. Auch die im Bereich der Rohhütte Helbra seit 1982/83 deponierten Theisenschlämme gefährden die Gewässergüte [SCHRECK 1997].

Darüber hinaus resultieren Beeinflussungen des hydrologischen Regimes unter anderem aus der Nutzung der Grundwasserressourcen zur Trinkwassergewinnung und aus Abwassereinleitungen in die Oberflächengewässer. Das Einzugsgebiet der SALZA enthält 10 Trinkwasserschutzgebiete mit insgesamt 4400 Hektar Fläche. Zwei Wasserwerke sowie 8 Pumpstationen gewährleisten die lokale Trinkwassergewinnung. Dabei wird der Bedarf an Trinkwasser nur zu 15% aus dem lokalen Angebot und zu etwa 85% durch Fremdwasser aus der Rappbode-Talsperre gedeckt, das dann dem Vorflutersystem als Abwasser wieder zugeführt wird [MIDEWA 2003]. Die meisten Gemeinden des Einzugsgebietes der SALZA organisieren die Beseitigung kommunaler Abwässer in fünf Abwasserzweckverbänden. Durch die steigende Zahl kommunaler Kläranlagen konnte der Anschlussgrad der Bevölkerung seit 1989 um rund ein Drittel erhöht werden. Darüber hinaus verringerte sich die Abwassereinleitung aus Industrieklärанlagen im selben Zeitraum um über 50% [STAU 1998 a].

Die SALZA lässt sich ökologisch *„... bisher nur in wenigen Abschnitten als intakt bezeichnen ...“* [SCHANZE 1998, S. 6]. Um die Mansfelder Seen als Bestandteil des Naturhaushalts zu sichern, wurde durch das Regierungspräsidium Halle (ehemals STAATLICHES AMT FÜR UMWELTSCHUTZ - STAU) der „Bewirtschaftungsplan SALZA“ (1995 bis 2002) aufgestellt. Er dient der Ordnung von Nutzungen des Gewässerregimes in ihrem Einzugsgebiet, der Zusammenführung der Daten zur biologischen (**Abb. 14**), chemisch-physikalischen und morphologischen Gewässerbeschaffenheit hinsichtlich geeigneter Bewirtschaftungsmaßnahmen und deren Bewertung auf Basis eines parameterbezogenen Zielsystems zur

angestrebten Umweltqualität. Die im Rahmen dieses Bewirtschaftungsplanes und der standardisierte Kontrolle der Gewässerbeschaffenheit erhobenen Daten liegen daher in zeitlich und räumlich hoher Dichte vor.

Ein wesentlicher Teil der Datenbasis dieser Arbeit bestand aus:

- den Geometriedaten zum Einzugsgebiet der SALZA (Gewässernetz, Messnetz, Teileinzugsgebiete, Siedlungen, Landnutzung, **Abb. 14 - Abb. 17**) [PFÜTZNER 1996]
- und den Ergebnissen biologischer sowie chemisch-physikalischer Untersuchungen zur Güteüberwachung von Fließgewässern im Regierungsbezirk Halle für den Zeitraum 1992 bis 1996 (**Abb. 15 - Abb. 17**) [STAU 1998 b].

Letztere halten die momentane Wasserbeschaffenheit an einem bestimmten Ort fest, liefern exakte Informationen über Art und Konzentration von Wasserinhaltsstoffen und sind eine Voraussetzung zur umfassenden Beschreibung der Gewässergüte. Parameterumfang und Untersuchungshäufigkeit variieren in Abhängigkeit von der Bedeutung des Gewässers und den spezifischen örtlichen Verhältnissen [STAU 1998 a].

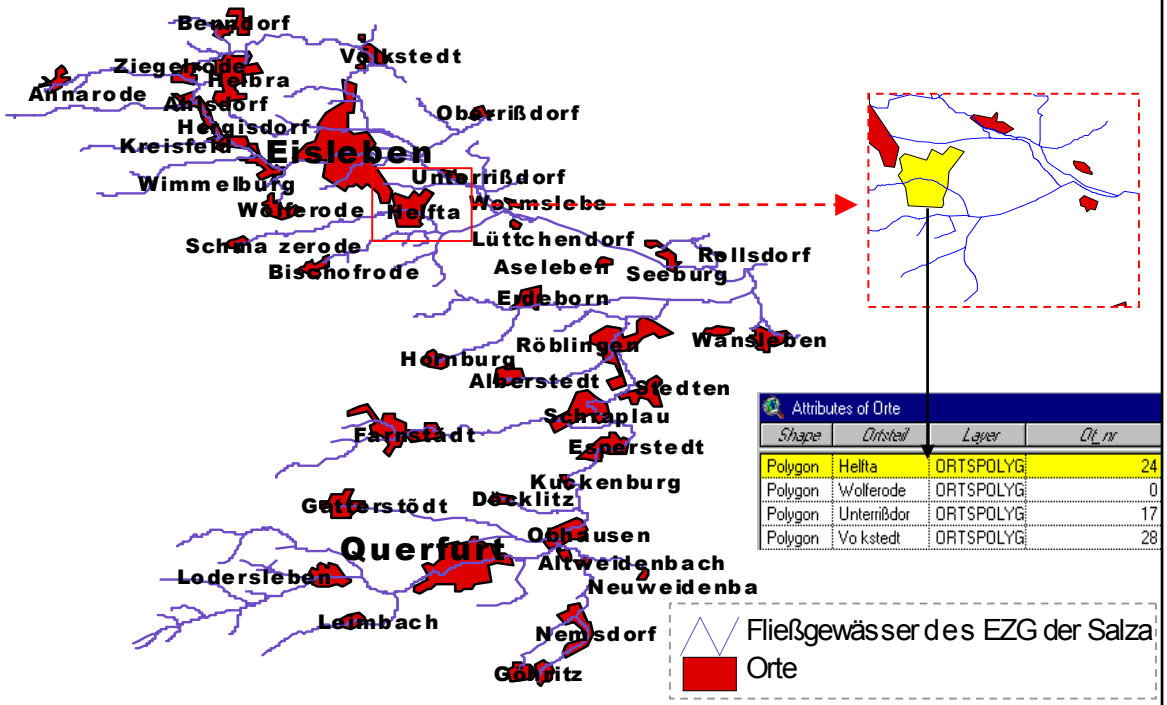


Abb. 14: Siedlungen im Einzugsgebiet der SALZA

Die biologische Gewässeruntersuchung spiegelt die Artenzusammensetzung der Biozönose im und am Gewässer wider und ermöglicht somit Langzeitaussagen. Es bestehen folgende Anforderungen an die Messungen:

- **Objektbezogene Messungen** (Einleiterüberwachung, **Abb. 17**): hier stammen die gemessenen Werte an der jeweiligen Messstelle überwiegend aus einer bekannten Emissionsquelle. Häufig werden gezielt, im Hinblick auf die Einhaltung von Grenz- und Richtwerten, bestimmte Kenngrößen überwacht.
- **Objektfreie Messungen** (gebietsbezogene Messungen, **Abb. 15**): Neben Emissionsüberwachung ist eine Immissionsüberwachung erforderlich, um die verschiedenen direkten und diffusen Stoffeinträge und deren Auswirkungen im Gewässer verfolgen und beurteilen zu können. Die Immission an der betreffenden Messstelle ist repräsentativ

für einen größeren Gewässerabschnitt und spiegelt die Gesamtheit der punktuellen und diffusen Einträge wieder. Die objektfreie Messung hat die Erfassung des allgemeinen Gewässerzustands zum Ziel, welche Daten zur Gewässerbeschaffenheit für repräsentative Zeiträume erfordert, um aussagekräftige statistische Parameter wie Trends und Streuungen ableiten zu können [LAWA 2000]. Konzentrationswerte sind nur in Verbindung mit steuernden Größen (z. B. Abfluss, Retention, Retardierung, biologische Prozesse) sinnvoll interpretierbar. Bei hohen Abflüssen bzw. Hochwasserereignissen sind abfluss- bzw. ereignisgesteuerte Probenahmen erforderlich [DVWK 1996], [ZIERDT & SCHMIDT 1996].

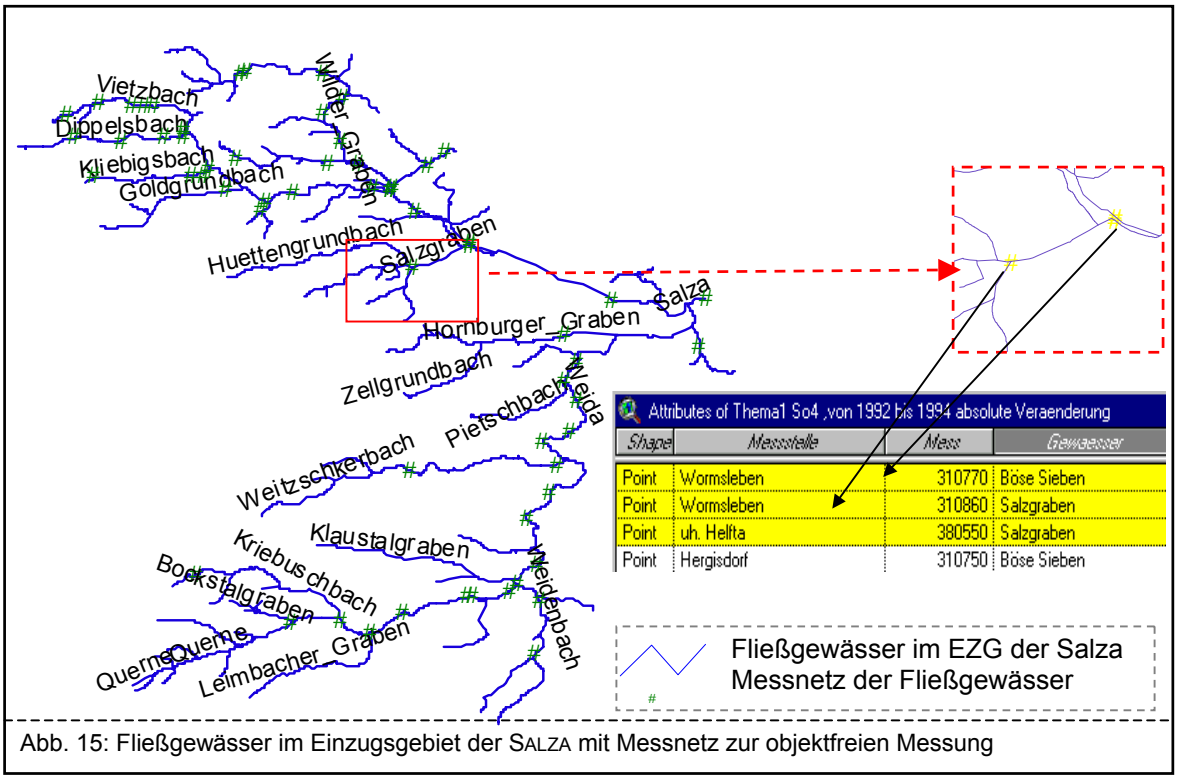


Abb. 15: Fließgewässer im Einzugsgebiet der SALZA mit Messnetz zur objektfreien Messung

Das Landesmessnetz im Regierungsbezirk Halle repräsentiert Fließgewässer mit landesweiter Bedeutung und besonderer wasserwirtschaftlicher Relevanz (Saale, Weiße Elster, Unstrut, Wipper, Helme). Im Regionalmessstellennetz befinden sich Gewässer von lokaler Bedeutung, wie zum Beispiel die SALZA, EINE oder RIPPACH. Bei Bedarf werden - zusätzlich zum Grundmessprogramm - weitere, den jeweiligen Bedingungen entsprechende spezifische Untersuchungen durchgeführt [STAU 1998 a].

Die Messnetze folgen Empfehlungen der LÄNDERARBEITSGEMEINSCHAFT WASSER [LAWA 2000], nach denen sie so eingerichtet werden sollten, dass alle im Untersuchungsgebiet bedeutsamen Fließgewässer erfasst werden und so regionale bzw. überregionale Zusammenhänge sowie anthropogene und geogene Belastungen erkennbar sind:

- vor der Einmündung wasserwirtschaftlich bedeutsamer Fließgewässer in Seen (bzw. Küstengewässer)
- ober- und unterhalb von Ortschaften und Industrieansiedlungen
- innerhalb wichtiger Fließgewässerabschnitte
- an anthropogen unbelasteten Flussabschnitten (Messstellen der Hintergrundbelastung, "Nullmessstellen").

Die Arbeitsschritte zur Aufbereitung und Hierarchisierung, Übertragung der Metadaten und anderer Attribute auf gewässerbezogene Datenschichten wie Teileinzugsgebiete,

Siedlungen oder Messnetz sowie die Nutzung sowohl in der Kopplung als auch in der Entkopplung von GIS und Datenbank sollten aufgrund ihrer hohen räumlichen und zeitlichen Dichte zunächst am Beispiel der vom Regierungspräsidium Halle zur Verfügung gestellten und hier kurz beschriebenen Daten zum Einzugsgebiet der SALZA entwickelt und erprobt werden. Darüber hinaus sollte sie jedoch auf jedes andere normalhierarchische Gewässernetz anwendbar sein (auch im Sinne einer Regionalisierung [DIEKKRÜGER 1999]), die z. T. wesentlich komplexer ausfallen können, wie beispielsweise das Gewässernetz der Tanger oder das der Unstrut.

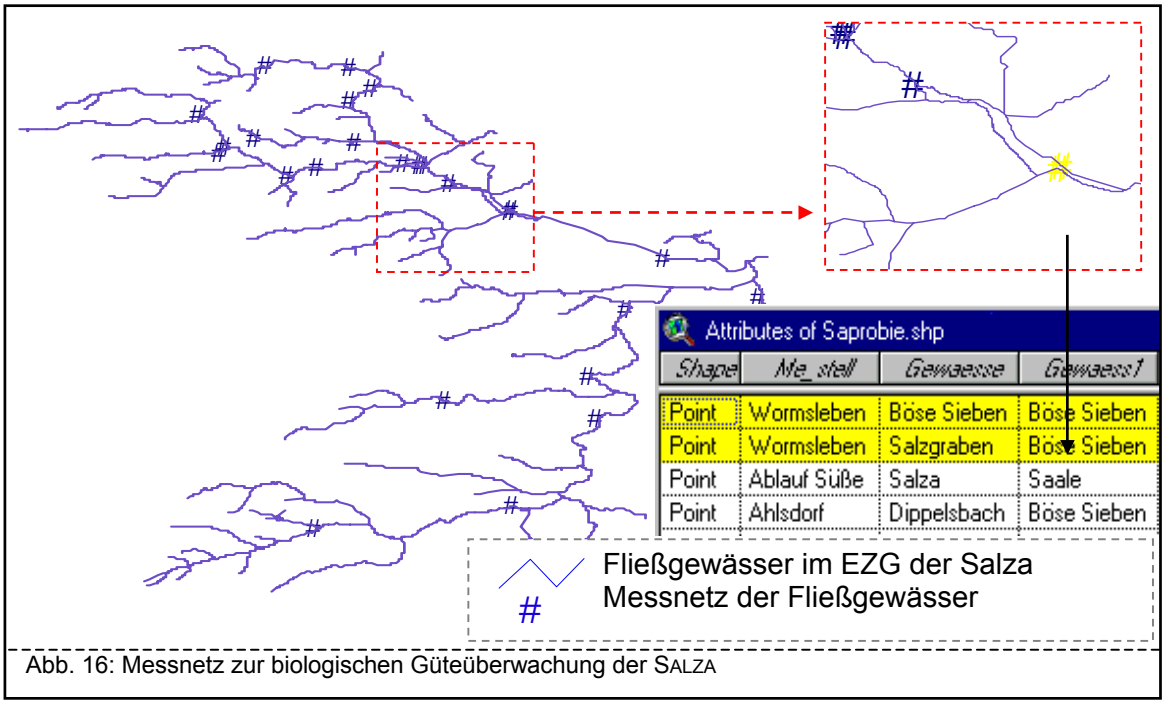


Abb. 16: Messnetz zur biologischen Güteüberwachung der SALZA

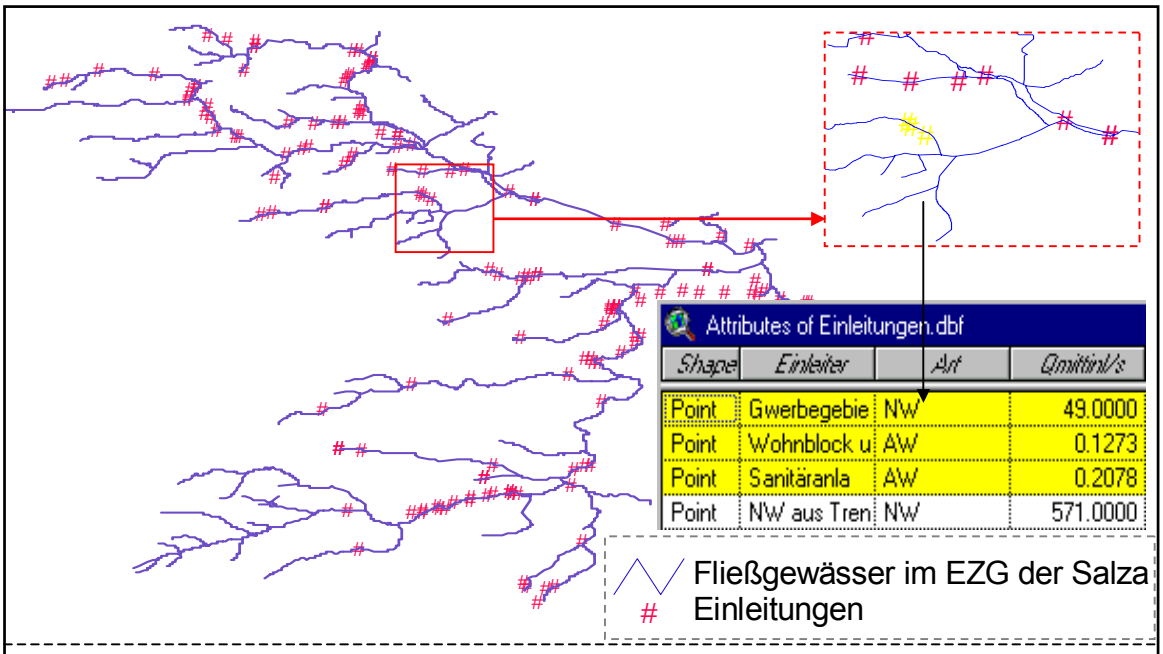


Abb. 17: Industrielle und gewerbliche Einleitungen im Einzugsgebiet der SALZA

Daher wurde die entstandene GIS-Erweiterung zur Verifizierung der Methodenentwicklungen neben dem Einzugsgebiet der SALZA auch an weiteren Gewässernetzen in verschiedenen Behörden und Institutionen getestet und konnte so an räumlich und inhaltlich unterschiedlichen Gebieten erprobt werden. Dies waren zum Beispiel:

- die digitale Fließgewässerkarte Dänemarks am DMU (Danske Miljø**u**ndersøgelser Instituten = National Environmental Research Institute (NERI)) in Selkeborg (**Abb. 18–1**) [Grundlage: DMU]
- Teile des Fließgewässernetzes von Brandenburg am LANDES-UMWELTAMT (LUA) Brandenburg (**Abb. 18 - 2**) [Grundlage: ATKIS]
- das Fließgewässernetz der Tanger und seiner Teileinzugsgebiete im Auftrag des LANDESAMTES FÜR UMWELTSCHUTZ (LAU) in Halle (**Abb. 18 - 3**) [Grundlage: ATKIS]
- das Fließgewässernetz der UNSTRUT und seine Teileinzugsgebiete (**Abb. 18 - 5**) im Rahmen des GLOWA (Globaler Wandel des Wasserhaushaltes) – Elbe – Projektes am POTSDAM INSTITUT FÜR KLIMAFOLGENFORSCHUNG (PIK) [Grundlage: TLUG, PIK]
- in der Bayerischen Landesanstalt für Wald und Forstwirtschaft, Forsthydrologie und Wasserhaushalt (SG II) das Einzugsgebiet der GROßEN OHE (**Abb. 18 – 6**).

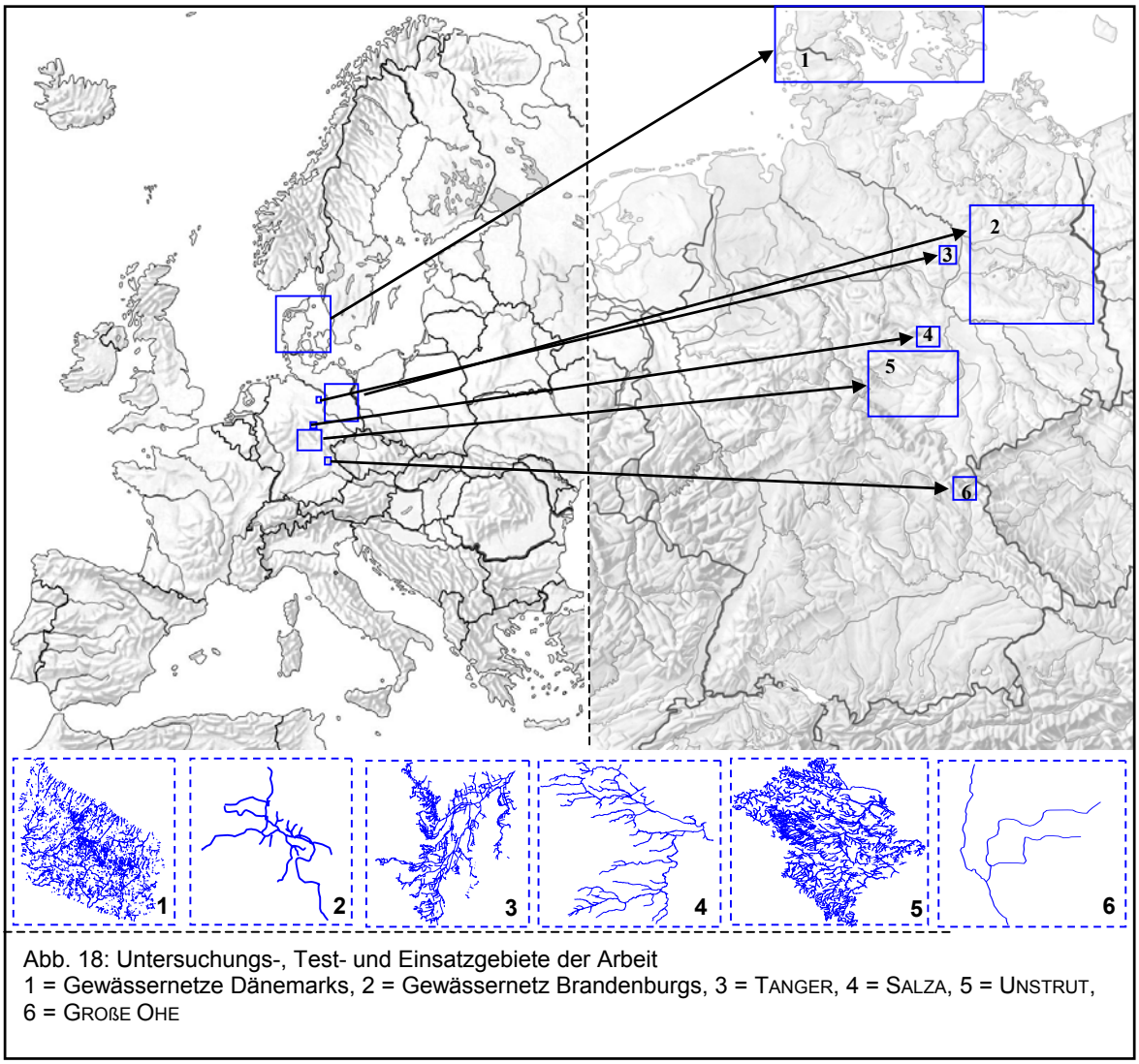


Abb. 18: Untersuchungs-, Test- und Einsatzgebiete der Arbeit
1 = Gewässernetze Dänemarks, 2 = Gewässernetz Brandenburgs, 3 = TANGER, 4 = SALZA, 5 = UNSTRUT, 6 = GROßE OHE

Durch diesen wiederholten Austausch von Ergebnissen der Arbeit und Datengrundlagen mit verschiedenen Institutionen und Firmen wurden die entwickelten und in Programmen umgesetzten Lösungsschritte mehrfach korrigiert, weiterentwickelt und zum Teil auch verworfen. Dabei traten über den gesamten Entwicklungszeitraum durch die unterschiedliche Perspektive und Ausgangsbasis der einzelnen Bearbeiter und Daten immer wieder neue Probleme, Hinweise und Verbesserungsvorschläge in den Entwicklungsprozess ein. So wurde beispielsweise die Analyse des Gewässernetzzusammenhanges sowie das Auffinden des Gebietsauslasses auf Anregung von Mitarbeitern des DMU in Selkeborg entwickelt, da dort besonders viele voneinander getrennte Gewässerbäume die Landoberfläche entwässern und eine automatisierte Bearbeitung auch in diesen Bereichen erhebliche Vorteile gegenüber der manuellen Korrektur bzw. Zuweisung hat. Auch das Gewässernetz der TANGER erwies sich als kompliziert und in Hinblick auf den Fortschritt der Arbeit besonders hilfreich. Die Funktionen zum Auffinden, Markieren und Speichern von Ringstrukturen wurden zum großen Teil an diesem Gebiet entwickelt. Die Darstellung der Problemlösungen und Teilergebnisse erfolgte daher auch an unterschiedlichen Testgebieten. Bei den Abbildungen handelt sich nicht um Kartendarstellungen im klassischen Sinne sondern um die Verdeutlichung der vorliegenden Problematik oder ihrer Lösung.

4 Ergebnisse

4.1 Hierarchieorientierte Präprozessierung digitaler Gewässernetze

4.1.1 Einführung

> *"Hierarchy ... is a partially ordered set. In less austere terms, a hierarchy is a collection of parts with ordered asymmetric relationships inside a whole. That is to say, upper levels are above lower levels, and the relationship upwards is asymmetric with the relationships downwards" [AHL & ALLEN 1996, S. 26].*

Die Behandlung von Hierarchien gewinnt in der heutigen Zeit immer mehr an Bedeutung. Unter dem Schlagwort XML wird vielfach über Implementierungen von XML-Datenbanken gesprochen, die im Grunde eine Renaissance hierarchischer Datenbanken sind [HEUER 1997]. Hierarchien werden jedoch in den meisten Datenbanksystemen bisher nicht explizit unterstützt. 1:n-Beziehungen können über Primär- bzw. Fremdschlüssel-Konstruktionen abgebildet werden. Sobald jedoch auf denormalisierte Tabellen Hierarchien aufgebaut werden sollen, sind Metadaten notwendig. Die hier beschriebenen Ansätze, Hierarchien in ein Datenbanksystem einzuführen, beruhen auf der Erweiterung relationaler Datenbanken um hierarchische Metadaten, die zur semantischen Query-Optimierungen und verbesserten Datenspeicherung im Umfeld eines Fließgewässermanagements beitragen sollen (vgl. **Kap. 4.3.1**). Ziel ist es also, den Sprachschatz des Standard-SQL für diese Zwecke auszunutzen, ohne auf spezielle, DBMS-abhängige Erweiterungen angewiesen zu sein.

⇨ Hierarchisierung im Sinne dieser Arbeit ist die Fixierung der hierarchischen Beziehungen innerhalb eines baumstrukturierten Gewässernetzes in Form von Attributen.

Drei Variante hierarchischer Abfrage sollten implementiert werden, die sich sowohl in ihrer Umsetzung sowie den ihnen zugrunde liegendenden Analysen und auf ihnen basierenden Lösungsmöglichkeiten verschiedenster fachspezifischer und allgemeinerer Fragestellungen unterscheiden:

- die aufwärtsgerichtete Abfrage
- die abwärtsgerichtete Abfrage
- die Abfrage der direkten Verbindung entlang des Gewässers (ab- und aufwärts)

Dabei wurde von unterschiedlichen Standpunkten ausgegangen und untersucht, welche Arten der Codierung sich finden lassen, inwieweit sie unter welche Bedingungen für verschiedene Fragestellungen relevant sind und wodurch sie sich sowohl in der Implementierung als auch der Nutzung auszeichnen.

Leider befinden sich digitale Daten sehr häufig nicht in einem für die automatische Extraktion hierarchischer Metadaten geeigneten Zustand. Die Aufnahme – Digitalisierung – lässt Raum für verschiedene Fehlerquellen wie z. B. nicht geschlossene Abschnittsenden, falsche Digitalisierungsrichtung, Überschneidungen usw. Auch die Weiterverarbeitung von Geodaten in Form räumlicher Operationen wie Verschneidung oder Pufferung bzw. ihre Ableitung aus Digitalen Höhenmodellen (DHM) haben häufig negative Auswirkungen auf die geometrische Integrität der Daten. *„The stream ordering processes may become a not difficult but time consuming process especially on large basins, large scale studies and dense networks. The use of Vektor GIS methodologies to deal with stream network studies introduces the possibilitiy of making use of computer facilities to solve the question"* [BURTON 1995, S. 139]. Um geometrische Daten zu Fließgewässern in einen für die

Hierarchisierung geeigneten Zustand zu überführen, stand am Anfang der Arbeit die Suche nach Möglichkeiten der weitgehend automatisierten Aufbereitung von Gewässernetzen und deren Realisierung in Programmansätzen innerhalb eines GIS. Zunächst stellt sich jedoch die schon in **Kap. 1.3** formulierte Frage: Welche Bedingungen muss ein digital vorliegendes Fließgewässernetz erfüllen, um hierarchisiert werden zu können und wie lassen sich diese Bedingungen weitestgehend automatisiert schaffen?

4.1.2 Analyse des geometrischen Zustandes digitaler Gewässernetze

Grundlage digitaler Gewässernetze innerhalb Deutschlands sind in den meisten Fällen, sofern sie nicht selbstständig digitalisiert oder abgeleitet wurden, die Geobasisdaten zu Stand- und Fließgewässern des Amtlichen Topographischen Informationssystems (ATKIS) des jeweiligen Landesamtes für Landesvermessung und Datenverarbeitung (LLVERMD). Die ATKIS-Daten, wie auch andere Vektordaten, beruhen auf der Digitalisiervorschrift des LLVERMD, sind jedoch häufig in verschiedenen Aspekten lückenhaft, wie z. B. den folgenden:

- Durchlässe und Düker
- verrohrte Abschnitte (z. B. in Stadtgebieten, Mündung seitlicher Zuflüsse)
- Gräben, Kanäle
- fehlende Abschnitte

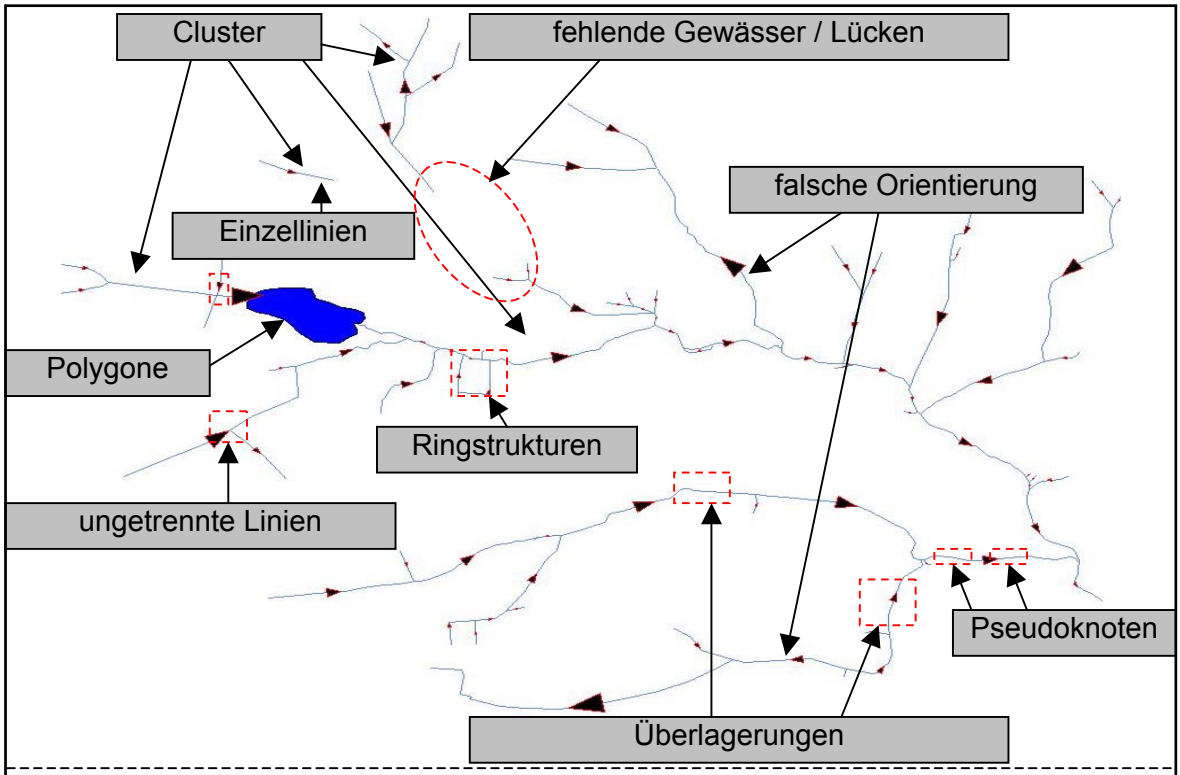


Abb. 19: Auswahl häufiger Fehler im geometrischen Datenbestand (Ausschnitt aus TANGER-EZG)

Darüber hinaus weisen digitalisierte Vektordaten eine Reihe von Fehlern und Einschränkungen auf, unter denen die folgenden wegen ihrer Häufigkeit von besonderer Bedeutung sind (**Abb. 19**):

- Die Gewässerstrecken besitzen keine Orientierung, wurden regellos digitalisiert. Damit ist die Fließrichtung aus den Daten nicht ableitbar.

- Überlagerungen der Gewässerabschnitte stellen eine schwer aufzufindende Fehlerquelle dar.
- Einzelne Abschnitte haben keine Verbindung zum Gewässernetz.
- Es sind in potenzieller Strömungsrichtung Lücken zwischen einzelnen Abschnitten vorhanden.
- Abschnitte sind falsch mit anderen Abschnitten verbunden. Die Folge können z. B. unerwünschte Kreisstrukturen sein (vgl. **Kap. 4.1.3.7**).
- Abschnitte kreuzen sich mit anderen Abschnitten.
- Es befinden sich Standgewässer in den Linienelementen. Auch hier muss eine Behandlung von Kreisstrukturen erfolgen.
- Die Bildung von Stationierungsachsen sowie die Kilometrierung (vgl. **Kap. 4.3.1**) sind nicht möglich.
- Die Benennung ist nicht immer einheitlich, teilweise auch falsch.
- Die Verschneidung mit den Teileinzugsgebieten (TGs) ergibt keine Übereinstimmung der Gewässerknoten mit den Gebietsauslässen.
- Die Länge der Streckenabschnitte bei der Objekt- bzw. Objektteilbildung entspricht nicht den fachlichen Anforderungen der Wasserwirtschaft.

Das Flussnetz muss jedoch zur weiteren Bearbeitung zumindest formal den Regeln eines vektorbasierten GIS entsprechen [BARTELME 1995]:

- Die Anfangs- und Endpunkte jeder Linie berühren entweder eine oder mehrere andere Linien oder sind mit Quellen identisch.
- Die Berührungspunkte zweier oder mehr Linien sind mit diesen Außenpunkten der Abschnitte identisch. D.h. an allen Schnittpunkten (Zusammenflüsse oder Verzweigungen) der Liniensegmente befinden sich gemeinsame Knoten.
- Die Stützpunkte[11] einer Linie sind von Start- zu Endpunkt geordnet, d.h. z. B. der dritte Punkt der Linie liegt auch räumlich zwischen dem zweiten und vierten.
- Jeder Punkt einer Linie – abgesehen von den Anfangs- und Endpunkten – ist innerhalb des Gewässernetzes nur einmal definiert. Es gibt also keine Überlagerungen von Abschnitten.

Jeder Abschnitt und jeder seiner Stützpunkte hat damit eine eindeutig zuweisbare Position innerhalb des Gewässernetzes.

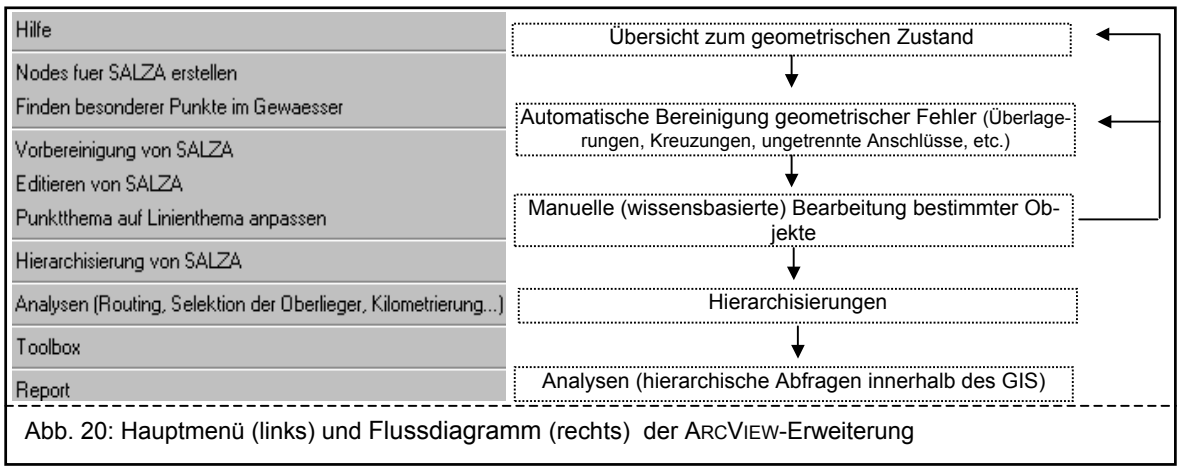


Abb. 20: Hauptmenü (links) und Flussdiagramm (rechts) der ARCVIEW-Erweiterung

[11] Die bei der Digitalisierung manuell oder automatisch erzeugten Punkte, aus denen sich die Linie zusammensetzt.

Um ein digitales Gewässernetz auf diese Kriterien hin möglichst effektiv untersuchen und ggf. korrigieren zu können, wurde eine ARCVIEW-Erweiterung (Extension) programmiert (**Abb. 20** links). Sie beinhaltet Funktionen zur Visualisierung des geometrischen Zustandes des Gewässernetzes, Extraktion topologisch wichtiger Punkte aus der Baumstruktur und zur eigentlichen geometrischen Optimierung, Analysen, Hierarchisierung, Selektion und manuellen Bearbeitung, auf die in folgenden Abschnitten noch näher einzugehen sein wird. Diese Funktionen orientieren sich im Interesse einer zielgerichteten Behandlung der digitalen Grundlagen in ihrer Anordnung an dem Bearbeitungsschema in **Abb. 1** und **Abb. 20**.

Alle Objekte eines Vektor-GIS lassen sich auf die Koordinaten der Punkte reduzieren, aus denen sie zusammengesetzt sind. Auch das Shapefile zu einem digitalisierten Gewässernetz besteht an seiner Basis aus solchen Koordinaten [ESRI 1998]. Da das Gewässernetz gewöhnlich auf topographischen Informationen wie Karten oder digitalen Höhenmodellen beruht und diese wiederum den natürlichen Verhältnissen nachempfunden sind, ist es sinnvoll, sich zu Beginn einen Überblick sowohl zu hydrologisch relevanten Punkten als auch Hinweisen auf eventuelle Fehlerquellen zu verschaffen. Zunächst stellt sich dabei die Frage, welche Arten punkthafter Informationen aus der Gewässergeometrie gewonnen werden können. BARROW [BARROW 1998] und andere Autoren bezeichneten Nodes als die wichtigste Art dieser Punkte. Oft wird dabei jedoch nicht weitergehend nach der Funktion unterschieden. **Tab. 4** gibt einen Überblick zu den in der Arbeit differenzierten (bis auf Überlagerungen) eindimensionalen Informationen.

Bezeichnung	**Beschreibung**
Stützpunkt (Vertice)	definiert die Geometrie ein- und zweidimensionaler geometrischer Elemente
Knoten (Node)	verbindet zwei oder mehr Linien miteinander, Untermenge von Stützpunkten
Echter Knoten	ist entweder Außenpunkt oder verbindet mehr als zwei Linien miteinander, Untermenge von Knoten
Pseudonode	verbindet genau zwei Abschnitte, Untermenge von Knoten
Quelle	berührt genau einen, aus ihm herausfließenden Abschnitt, Untermenge von Knoten
Außenpunkt	wird von genau einer Linie berührt, Untermenge von Knoten
Zusammenfluss	n (n > 2) Abschnitte berühren sich in diesem Punkt, wobei gilt: n-1 Abschnitte fließen in diesen Punkt hinein, genau ein Abschnitt fließt aus ihm heraus, echter Knoten
Verzweigung	n (n > 2) Abschnitte berühren sich in diesem Punkt, wobei gilt: n-1 Abschnitte fließen aus diesem Punkt heraus, genau ein Abschnitt fließt in ihn hinein, echter Knoten
Top	alle diesen Punkt berührenden Linien beginnen dort
Senke	alle diesen Punkt berührenden Linien enden dort
Anfangs- und Endpunkte	Untermenge von Knoten
Abschnittszentrum	geometrisches Zentrum der Linie
Unklare	die Anzahl der Zu- und Abflüsse aus diesem Punkt ist jeweils >= 2

Kreuzung	mindestens zwei Linien berühren sich über eine Distanz = 0, wobei keine Knoten beteiligt ist
Ungetrennte Abschnitte	das Ende/der Anfang einer Linie berührt einen Vertice, jedoch keinen Knoten
Überlagerung	mindestens zwei Linien berühren sich über eine Distanz > 0 (eindimensional)

Tab. 4: Wichtige Punkte im digitalen Gewässernetz

Auf Basis dieser Definitionen lässt sich der geometrische Zustand eines Gewässernetzes für den Bearbeiter relativ übersichtlich und intuitiv darstellen (**Abb. 21**). So sind z. B. Punkte, die als Quellen im resultierenden Thema angezeigt werden, entweder nicht korrekt angeschlossene Gewässerabschnitte, oder aber Quellen im eigentlichen Sinne. So kann man feststellen, wo Korrekturen (durch Snappen) vorgenommen werden müssen und wie viele Gewässerabschnitte betroffen sind. Sie bilden außerdem die Grundlage der automatisierten Extraktion der zugeordneten Punkte aus dem Gewässernetz, die im Folgenden erläutert wird[12]. Dabei werden für das Gewässernetz alle Arten von in **Tab. 4** definierten Punkten herausgefiltert, sofern sie dort vorhanden sind (**Abb. 21**).

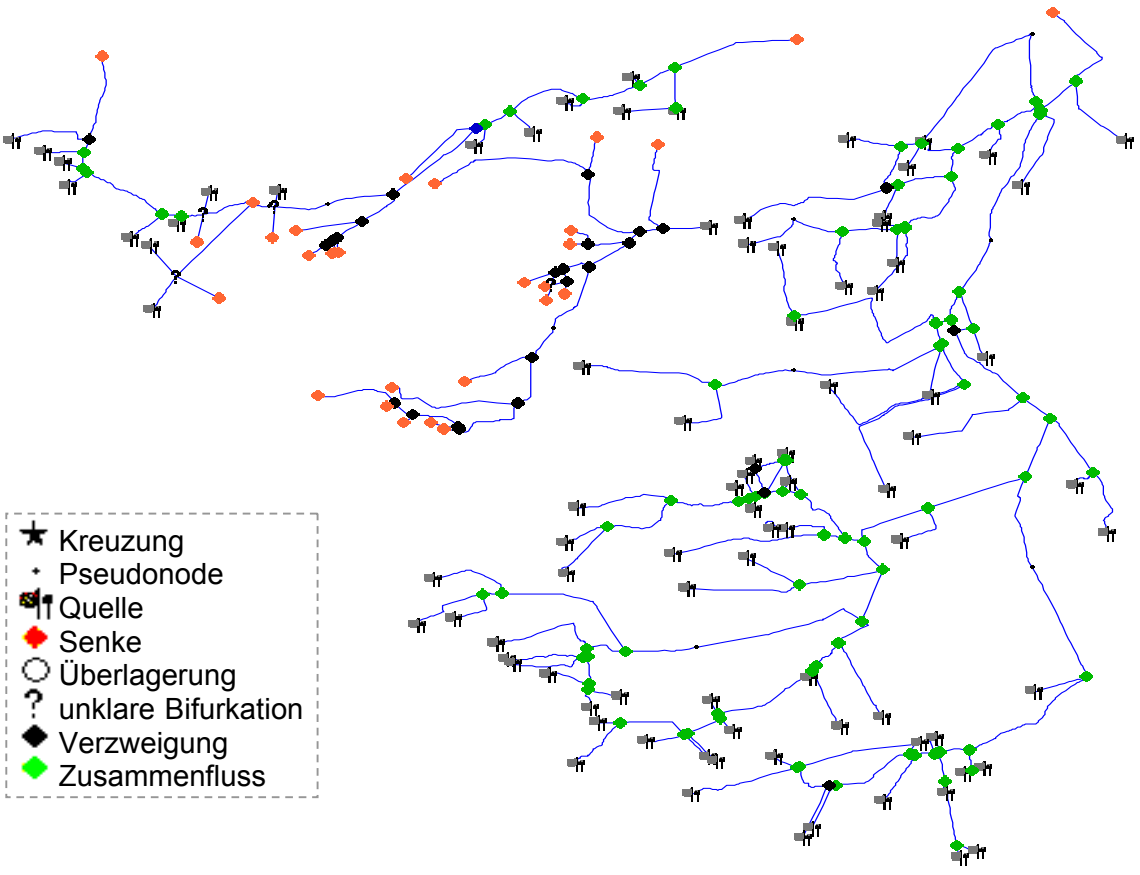
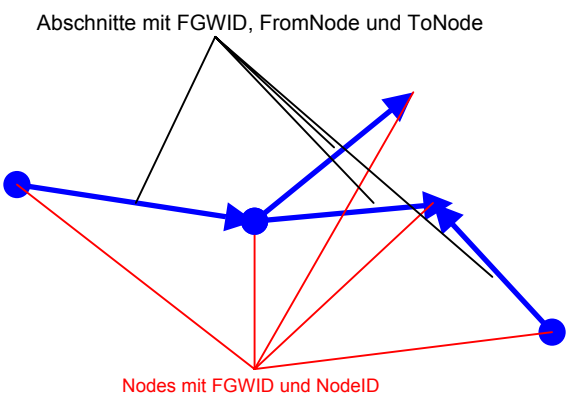


Abb. 21: Nodes - Überblick zum geometrischen Zustand des Gewässernetzes (TANGER, Ausschnitt)
Nur der rechte Gewässerbaum wurde bereits bereinigt, enthält jedoch Ringstrukturen. Die Legende (links) spiegelt die digitale Geometrie des Gewässernetzes wider.

[12]Bis auf sehr kurze Programmfragmente befinden sich die meisten Ausschnitte des eigentlichen Codes, zusammen mit Kommentaren zum Verständnis der Programmabläufe, im Anhang.

Dies geschieht folgendermaßen (vgl. **Code 2**): Jeder Abschnitt wird in seine Stützpunkte aufgelöst und einem Array (Datenstruktur, Liste) `vertices` hinzu gefügt. Ein Array `starts` erhält die Anfangspunkte, ein Array `ends` die Endpunkte und ein Array `centers` die geometrischen Zentren der Linien. Die Nachbarn des Abschnittes werden selektiert. Falls sich keine Nachbarn finden, wird der Abschnitt einem Array `singleLines` zugeordnet. Anfangs- und Endpunkt werden – sofern sie keinen der eventuell bereits im Array `nodes` befindlichen Punkte treffen – dieser Datenstruktur zugeordnet. Hier wird also darauf geachtet, dass keine Überlagerungen von Punkte in die Nodes-Liste gelesen werden. Außerdem wird die Anzahl der diese beiden Punkte berührenden Anfänge oder Enden der mit dem aktuellen Abschnitt selektierten Linien gezählt. Bei einer gefundenen Intersektion liegt ein Außenpunkt vor. Zwei definieren einen Pseudonode und mit dreien wurde eine Bifurkation gefunden. Sollte ein Außenpunkt den Anfang des aktuellen Abschnittes berühren, so ist dieser eine Quelle. Eine Bifurkation wird mithilfe eines Vergleiches der Anzahl von Nachbarn mit der Anzahl von Anfängen dieser Nachbarn, die diese Bifurkation berühren, daraufhin untersucht, ob sie Zusammenfluss oder Verzweigung ist. Am Ende werden die gefüllten Arrays an das aufrufende Skript übergeben und dort ausgewertet[13].



Fgwid	Fromnode	Tonode
13	1	2
14	3	1
15	4	1
21	5	6
31	7	8
46	9	10
152	10	11

ID	X	Y	Art
1	4459841.50000000	5713435.00000000	Zusammenfluss
2	4463588.50000000	5713048.00000000	Senke
3	4459356.57841237	5712783.26207035	Quelle
4	4458432.91444433	5712803.15577687	Quelle
5	4457083.06195177	5711836.50746031	Quelle
6	4463348.00000000	5712340.00000000	Senke
7	4459877.69910360	5710633.78347294	Quelle
8	4464286.50000000	5710962.50000000	Senke

Abb. 22: Attributive Verbindung von Abschnitten und Knoten: die obere Tabelle bezieht sich auf die Gewässerabschnitte, die untere Tabelle auf die Gewässerknoten

Mithilfe von **Code 2** lassen sich in einer Iteration (Durchlauf) über alle selektierten Datensätze eines Linienthemas die in **Tab. 4** definierten Informationen aus der zweidimensionalen Geometrie extrahieren. Der offensichtliche Vorteil liegt in der Geschwindigkeit und Flexibilität des Einsatzes. So kann dieser Algorithmus bei der Qualitätskontrolle in Form eines klassifizierten Punktthemas (**Abb. 21**), zur Erzeugung einer graphischen Übersicht oder zur Speicherung eines bestimmten Typs der extrahierten Information als separater Layer verwendet werden.

Ein Nodethema[14], wie es in **Abb. 21** gezeigt wurde, behält dabei die Verbindung zum Gewässernetz über die ggf. hinzugefügten Attribute FGWID (Fließgewässer-ID, eindeutige Kennung), FROMNODE (Anfangspunkt) und TONODE (Endpunkt) im Linienthema und FGWID im Nodethema, die dem ARC/INFO-Datenmodell entlehnt sind. Die Ausrichtung eines Abschnittes wird über die Attribute FROMNODE und TONODE beschrieben (vgl. **Kap.**

13 Um das Zentrum einer Linie zu erhalten, gibt es in Avenue einen Befehl `returnCenter`, in Visual Basic wurde dazu eine Funktion geschrieben, der eine Linie und eine Prozentzahl übergeben werden – für das Zentrum also eine 50.
14 Thema = Layer, geographische Informationsschicht

4.1.3.1, **Abb. 22**). Knoten können über ihnen zugeordnete Attribute zusätzliche Informationen wie die Höhenlage aufnehmen und damit z. B. zur (groben) Charakterisierung der Gefälleverhältnisse im Gewässer beitragen. Ihnen können aber auch Informationen zu Bauwerken, Messeinrichtungen usw. zugewiesen werden.

Diese Funktionen dienen also dem ersten Überblick, aber auch der immer wieder möglichen Kontrolle des Zustandes des Gewässernetzes (vgl. **Abb. 21**). Nach Bearbeitungsschritten, welche die Nachbarschaftsbeziehungen des Gewässernetzes verändern, wird der zum Gewässernetz gehörende Nodelayer optional aktualisiert.

Unter den in **Tab. 4** beschriebenen Informationstypen fällt das Auffinden von Überlagerungen (**Abb. 23**) durch die Eindimensionalität der Problemstellung aus dem Rahmen und ist deshalb mit **Code 4** dargestellt. Solche Überlagerungen sind typisch für ein nichttopologisches System und die Quelle verschiedener Irritationen beim Bearbeiter und von Fehlern bei räumlichen Operationen wie Verschneidung, Pufferung oder Abfragen nach der Lage. Auch eine Hierarchisierung wäre dadurch gefährdet. Sie können bereits bei der Digitalisierung entstehen oder durch unkorrekte Verschneidung und sind normalerweise nicht sichtbar. Ein solcher Test sollte daher nach Manipulationen am Datenbestand (wie z. B. Verschneidungsoperationen) durchgeführt werden. Im Nodethema werden die Zentren der Überlagerungen abgebildet. Dies ermöglicht zunächst die visuelle Zuordnung und eine Selektion der betreffenden Abschnitte.

Überkreuzungen (**Abb. 24**) und nichtangeschlossene Abschnitte (**Abb. 25**) stellen ein anderes häufig auftretendes Problem dar. Auch sie können bei der weiteren Arbeit mit dem Datenbestand störend wirken und müssen bereinigt werden.

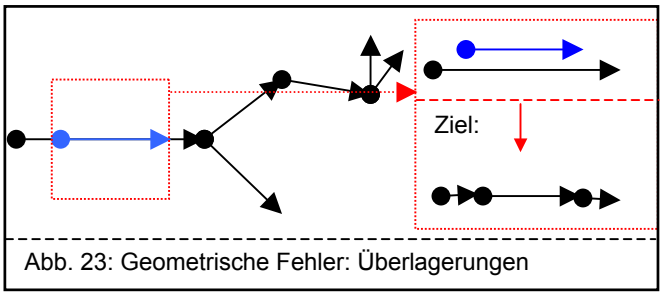


Abb. 23: Geometrische Fehler: Überlagerungen

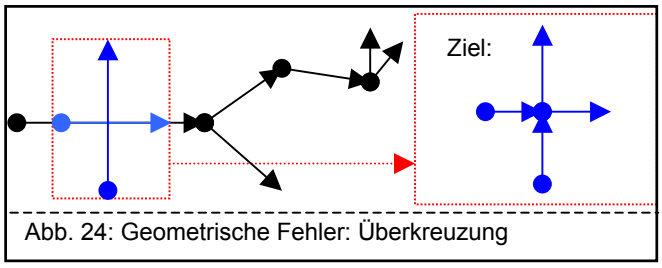


Abb. 24: Geometrische Fehler: Überkreuzung

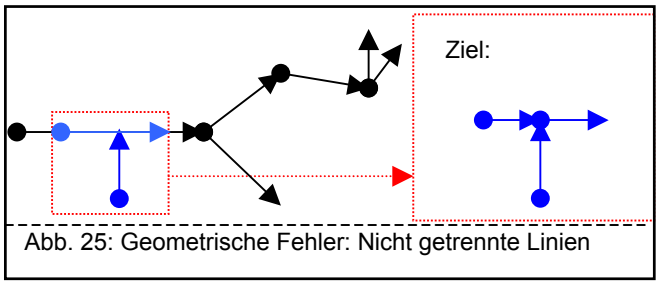


Abb. 25: Geometrische Fehler: Nicht getrennte Linien

Das Vorgehen ähnelt dem des Auffindens von Überlagerungen. Durch eine Kontrolle der Berührung einer Linie mit anderen Linien im selben Layer wird für jeden Abschnitt eine Vorauswahl der zu untersuchenden Nachbarschaftsbeziehungen getroffen. Dabei wird festgestellt, ob der Start- oder der Endpunkt eine andere Linie an deren Start- oder Endpunkt berührt. Ist dies nicht der Fall und liegt auch keine Überlagerung vor, so ist der mit der oder den anderen Linie(n) gemeinsame Punkt eine Überkreuzung.

Hat der betrachtete Abschnitt einen Nachbarn, den er mit seinem Start- oder Endpunkt an einem Vertice, jedoch nicht an einem Node berührt, so liegt an dieser Stelle eine nicht getrennte Linie vor (vgl. **Tab. 4**, **Abb. 25**).

Die Legende wird gleichzeitig erzeugt und gespeichert (vgl. **Abb. 21**). Dies vereinfacht die automatisierte Darstellung der Klassifikation aus **Tab. 4** und die Weitergabe dieser Information z. B. an andere Bearbeiter, Institutionen oder Firmen, zusammen mit dem Nodethema und dem Gewässernetz.

Dieser Bearbeitungsschritt dient also der Übersicht zum geometrischen Zustand des Gewässernetzes. Die Korrektur eventuell vorhandener Fehler wird in den nächsten Kapiteln besprochen.

4.1.3 Korrektur von Fehlern im geometrischen Datenbestand

4.1.3.1 Einführung

Der Automatisierungsgrad dieser Korrektur von Fehlern im Gewässernetz hängt sehr von den speziellen Verhältnissen im betrachteten Gewässernetz ab. So können Einzellinien Fehler sein oder existierende Fließgewässer abbilden, die z. B. auf kalkigem Untergrund versinken und damit nicht ober-, sondern unterirdisch entwässern. Ringstrukturen können durch Kanäle geschlossen sein oder aber durch eine falsche Digitalisierung entstehen. Abschnitte, die nur durch Pseudoknoten voneinander getrennt sind, sollten möglicherweise, aber nicht zwingend, zusammengefasst werden. Auch hier ist das Spezialwissen des Bearbeiters gefragt. Im Laufe der Entwicklung und im kontinuierlichen, kritischen Austausch mit Institutionen (vgl. **Kap. 1.1**, **Kap. 1.3**), bei denen sich dieses Werkzeug im Einsatz befand, wurde eine interaktive, zum großen Teil automatisierte Bearbeitungsreihenfolge entwickelt (vgl. **Abb. 1** und **Abb. 20**). Die hier verfügbaren Funktionen sind:

- Einzellinien (anzeigen und ggf. entfernen)
- Pseudoknoten (anzeigen und ggf. entfernen)
- Lücken (schließen)
- Überlagerungen (anzeigen und ggf. entfernen)
- Kreuzungen und nicht getrennte Anschlüsse (aufsplitten)
- Fließrichtung (korrigieren)
- Ringstrukturen (anzeigen und ggf. separat speichern)
- Cluster (Linienzusammenhänge anzeigen und speichern)

Teile dieses als Sukzession angelegten und mehrfach erprobten Bearbeitungskonzeptes werden auch bei der Analyse des geometrischen Zustandes eingebunden, sofern solche Problemstellen im Gewässernetz gefunden wurden (vgl. **Kap. 4.1.1**). Dadurch ist ein höherer Grad der Automatisierung des Präprozessings möglich. Ohnehin durchzuführende Arbeitsschritte laufen im Hintergrund ab, von der Benutzeroberfläche getrennt. Auf diese Art wird die Analyse an eine Vorbereinigung des Gewässernetzes gekoppelt und die Zahl der Arbeitsschritte für den Benutzer auf ein Minimum reduziert. Danach sind nur noch das Schließen eventuell vorhandener, unerwünschter Lücken und die Hierarchisierung nötig.

Die Arbeitsschritte können jedoch auch nacheinander kontrolliert vollzogen werden. Dabei werden die gesuchten Fehlerquellen zunächst als Grafiken (Symbole) angezeigt, wodurch sich die Möglichkeit ergibt, einzelne Objekte nach bestimmten Kriterien von der weiteren Bearbeitung auszuschließen, indem man die zugeordneten Grafiken entfernt. Neben der Ergonomie hat diese Verfahrensweise den Vorteil, dass die Algorithmen zum Auffinden der Fehlerquellen nur einmal, nämlich zur Erzeugung der Grafiken ablaufen müssen. Die nachfolgenden Programmabläufe stützen sich auf diese Ergebnisse. Im Detail ist dann meist nur noch eine Selektion der betreffenden Datensätze mit den Grafiken erforderlich –

eine einfache räumliche Operation. Dementsprechend lag ein Schwergewicht der Überlegungen zum Entwurf und zur Umsetzung dieses Teilbereiches der Applikation überwiegend im Auffinden der genannten Fehler (z. B. **Abb. 27**). Auch hier kann auf die dahinter stehenden Programme nur auszugsweise eingegangen werden.

4.1.3.2 Einzellinien und Pseudoknoten

Berührt weder Anfangs- noch Endpunkt einer Linie einen anderen Gewässerabschnitt so liegt eine separate Linie vor. **Code 2** ging bereits darauf ein (**Abb. 26**). Diese müssen und können ggf. ausgewiesen, gespeichert oder entfernt werden.

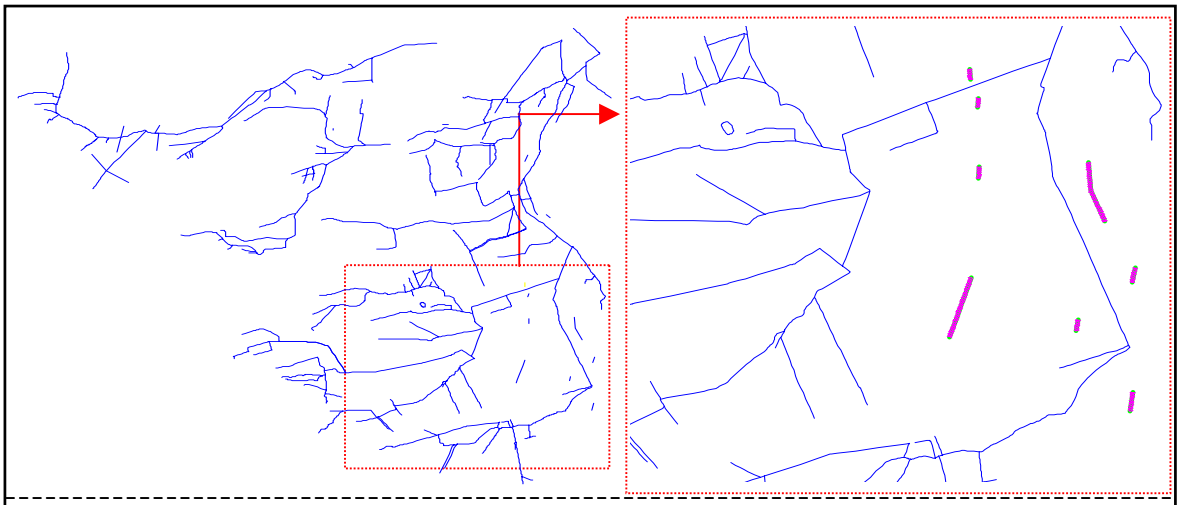


Abb. 26: Graphische Darstellung der gefundenen Fehlerbereiche (in diesem Fall fett dargestellte rote Einzellinien, TANGER, Ausschnitt)

Auch das Auffinden von Pseudoknoten ist in **Code 2** beschrieben. Da es nicht in jedem Fall erwünscht ist, die Trennung von Abschnitten aufzuheben, kann die Zusammenführung von Abschnitten auf Basis bestimmter Attribute gesteuert werden. Sofern diese nicht vollständig übereinstimmen, bleiben die betreffenden Linien getrennt. Auf diese Art ist es möglich, beispielsweise in den Datenbestand hinein digitalisierte Düker oder Kanäle vor einer Zusammenführung mit normalen Gewässerabschnitten zu bewahren (vgl. **Abb. 28**).

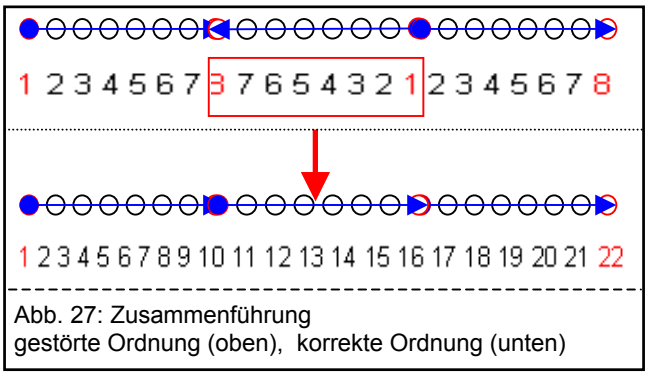


Abb. 27: Zusammenführung
gestörte Ordnung (oben), korrekte Ordnung (unten)

Wie in **Kap. 4.1.2** beschrieben, muss eine geometrisch korrekte Linie aus einer geordneten Reihe Vertices vom Anfangs- bis zum Endpunkt bestehen. Bei der Verbindung mehrerer Linien werden die Stützpunkte nicht automatisch neu geordnet; problematisch vor allem dann, wenn sich viele, z. B. mehrere hundert Abschnitte zwischen zwei echten Knoten befinden (**Abb. 27**). **Code 4** zeigt einen Lösungsweg auf. Hier wird zunächst eine der beiden Außenlinien gefiltert ggf. ausgerichtet. Von dieser Startlinie ausgehend durchsucht das Programm die verbleibenden Glieder der Kette nach dem Nachbarn, der ggf. gedreht und im nächsten Durchlauf als Startlinie und in das Ausgangsarray an Stelle der falsch orientierten Linie eingesetzt wird. Dies geschieht solange,

bis das andere Ende erreicht ist. Dieser Algorithmus kommt u. a. auch bei der Bereinigung von Überlagerungen sowie der Kilometrierung (vgl. **Kap. 4.3.1**) zum Einsatz.

4.1.3.3 Lücken

Die Überprüfung der Nachbarschaft dient auch der Zusammenführung von Abschnitten, die durch Lücken getrennt sind. Lücken entstehen ebenfalls bei der Digitalisierung oder bei räumlichen Operationen wie Verschneidung oder Clippen (Ausschneiden) und sind oft auf den ersten Blick nicht zu erkennen. Sie können zwar automatisiert erfasst, jedoch nicht ohne das Fachwissen des Bearbeiters geschlossen werden. Das Problem für den Bearbeiter besteht darin, diejenigen Abschnitte zu finden, die fälschlicherweise räumlich getrennt vorliegen. Einschränkungsmerkmal sind die maximale Distanz, innerhalb der nach Lücken im Gewässernetz gesucht wird, sowie Attribute, die zusammengehörende Abschnitte ausweisen. Die Suche basiert auf zwei mögliche Anschlusskriterien, die vorgegeben werden können (**Abb. 29**):

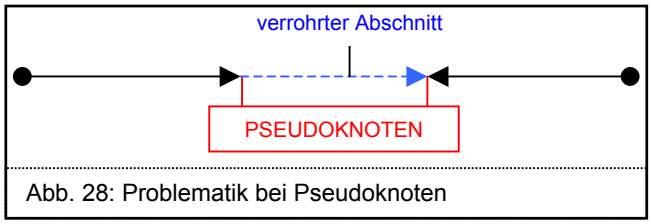


Abb. 28: Problematik bei Pseudoknoten

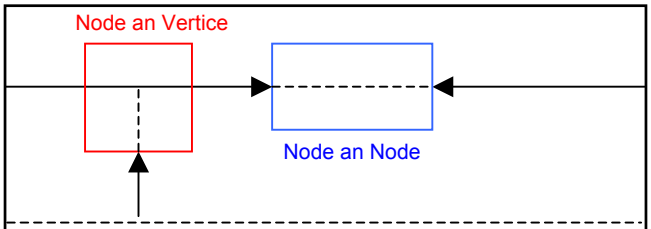


Abb. 29: Problematik bei Nachbarschaft, der zu korrigierende Bereich ist gestrichelt dargestellt

- Node an Node
- Node an Vertice

Beim Anschluss an einen Stützpunkt muss zusätzlich ein neuer Knoten in das System eingefügt werden. Um bei der Aktion „Node an Vertice“ die geometrische und inhaltliche Integrität der aufgesplitteten Linien zu wahren, wird an das Ende der einen und an den Anfang der anderen entstehenden Linie der Endpunkt der dorthin verlängerten Linie platziert und die Attribute der Ausgangslinie auf die beiden Segmente übertragen.

4.1.3.4 Überlagerungen

Überlagerungen werden durch eine Verschneidung der beteiligten Linien untereinander und die Aktualisierung der Datenbasis mit den entstehenden Teillinien bereinigt.

4.1.3.5 Kreuzungen und ungetrennte Linien

In der Arbeit wurde die Flexibilität gegenüber dem ARC/INFO-Modell[15] insofern erhöht, dass der Bearbeiter festlegen kann, ob – und wenn, nach welchen Kriterien – nur ein Teil der aufgesplitteten Abschnitte in das Gewässernetz integriert werden soll. **Abb. 30** verdeutlicht diese Möglichkeiten. Die dabei entstehenden neuen Abschnitte erhalten die Attribute der Ausgangslinien.

[15] ARC/INFO hält auch hier mit den Befehlen `build` und `clean` eine einfache Lösung des Problems bereit, bei der alle beteiligten Linien an diesen Kreuzungen aufgespaltet und die dabei neu entstehenden Abschnitte an die Attributtabelle angehängt werden.

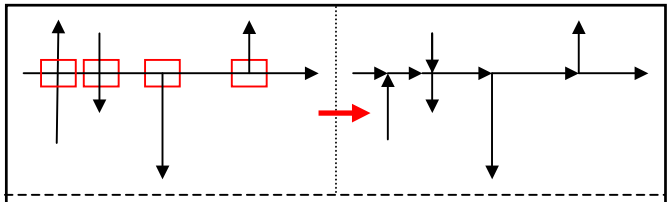


Abb. 30: Kreuzungen und ihre Korrektur

Nach der Beschreibung der geometrischen Korrekturen befassen sich die folgenden Abschnitte mit globalen Aspekten der Aufbereitung und Analyse eines Gewässernetzes, zu denen die Ausrichtung, der Gewässerzusammenhang, das Auffinden von Ringstrukturen und des Gebietsauslasses sowie die Integration von Seen oder Flüssen mit Doppellinien gehören. Dies entspricht auch der erarbeiteten Empfehlung zur Reihenfolge der hierarchieorientierten Aufbereitung von Fließgewässern. Die Hierarchisierung und ihre Nutzung im Gewässermanagement schließen sich daran an und werden in späteren Kapiteln erläutert.

4.1.3.6 Fließrichtung

Zu diesem Zeitpunkt haben die Liniensegmente noch keine Baumstruktur. Sie sind einzeln abgespeichert, ohne Informationen über Flussverzweigungen, Fließrichtungen oder Flusslängen. Daher bieten einige GIS mehr oder weniger generalisierte Routinen an, die jedoch nicht speziell auf die Anforderungen bei der Arbeit mit Flussnetzen abgestimmt sind. Das Prinzip der Linienvernetzung beruht meist darauf, ausgehend von einem Knoten über das dazwischenliegende Segment jeweils den nächsten Knoten zu suchen, solange, bis die gewünschten Verbindungen erreicht sind. Dabei werden jedoch oft weder Fließrichtungen definiert, noch kann beispielsweise zwischen Haupt- und Nebenflüssen unterschieden werden. Meist führen diese Verfahren bei großen Datensätzen auch zu langen Prozesszeiten [PAIVA & EGENHOFER 1997], [LEHNER 1998].

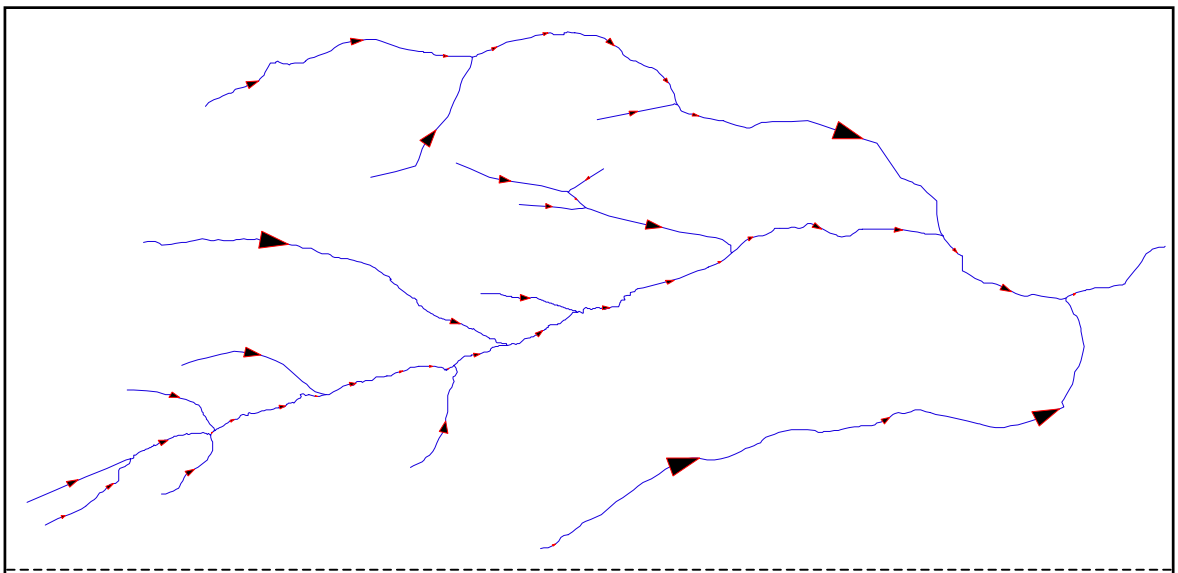


Abb. 31: Darstellung der Fließrichtung nach der Korrektur (QUERNE, Teil-EZG der SALZA)

Bei der Bearbeitung des Testgebietes der Unstrut waren etwa 25% der Linien aus den ATKIS-Datenbeständen falsch orientiert. Sollen nicht nur selektierte Abschnitte gedreht werden, kommt ein Algorithmus zum Einsatz, der auch beim Aufspüren von Ringstrukturen sowie in einigen Bereichen der Hierarchisierung wiederzufinden und auszugsweise in

Code 6 erläutert ist. In seiner reduzierten Version geht das Programm von der Wurzel den Gewässerbaum hinauf und dreht ggf. alle auf dem Weg zu den Quellen gefundenen Abschnitte. Die From- und ToNode-Beziehungen werden aktualisiert. Das Ergebnis sollte nach der Bearbeitung dem in **Abb. 31** dargestellten kleinen Gewässernetz entsprechen.

Um die hydrologischen Verhältnisse möglichst genau abzubilden, müssen natürlich auch Informationen wie Wasserscheidengrenzen, Convergent Flow (Talfluss) oder Divergent Flow (Kammabfluss) berücksichtigt werden können. Es besteht dabei zu jedem Zeitpunkt die Möglichkeit, die Ausrichtung der Abschnitte zu steuern, z. B. durch die schon beschriebene manuelle Richtungskorrektur oder durch die Verschneidung mit der Wasserscheide. Hierbei wird ein Stopp-Attribut belegt, das bestimmt, an welchen Abschnitten eine automatisierte Ausrichtung abgebrochen werden soll. Programmintern werden diese dann als Quellen behandelt.

4.1.3.7 Ringstrukturen

Auch das Auffinden von Ringstrukturen hat eine wissensbasierte Komponente, da nur dem Bearbeiter bekannt ist, welche Art von Kreisstruktur vorliegt. Ist es ein See, ein Rückhaltebecken, ein geschlossener Kanal, ein durch Melioration vernetztes Grabensystem oder das Resultat eines Digitalisierfehlers? Besonders im Flachland (z. B. TANGER-Einzugsgebiet) finden sich häufig solche Problembereiche. Sie entsprechen nicht den in **Kap. 2.3** definierten Kennzeichen einer Baumstruktur und ermöglichen daher keine automatisierte Hierarchisierung, es sei denn, diese beruht auf einer Vereinbarung mit dem Bearbeiter, in der z. B. festgelegt wurde, dass immer der kürzere Teil der Außenlinie einer Ringstruktur (oder z. B. der mit dem höchsten Durchfluss, sofern dieser bekannt ist) in das Gewässernetz integriert werden soll. Alle anderen Abschnitte könnten dann durch ein Attribut oder geometrisch ausgeschlossen werden.

Das Programm[16] beginnt am Ausgangspunkt – für gewöhnlich dem Gebietsauslass - und Untersucht das Gewässernetz schrittweise in Richtung der Quellen. Die jeweils stromauf liegenden Nachbarn eines Abschnittes werden ggf. gedreht und ihr Bearbeitungsstand in einem dafür angelegten Feld festgehalten. Treten im Gewässernetz Ringstrukturen auf, so liegt es auf der Hand, dass an irgendeiner Stelle in solch einem Ring der nächste gefundene Nachbar schon einmal bearbeitet worden sein muss (Einstiegspunkt in Rücklauf **Abb. 32**, vgl. auch **Kap. 4.2.5.2**). Innerhalb des Algorithmus wird zunächst geklärt, ob die Bewegung im Ring links oder rechts gerichtet ist (eine boolsche Variable `isRight` wird dabei temporär gespeichert, sie ist entweder wahr oder falsch). Eine Subroutine (vgl. **Kap. 4.2.5.2**) ermittelt daraufhin denjenigen Nachbarn am Einstiegspunkt, der entsprechend dieser Vorgabe ganz links oder ganz rechts von der zuletzt bearbeiteten Linie liegt und sich somit bereits im Ring befindet. Mit der Variablen `isRight` ist auch die Position des jeweils nächsten im Ring zu betrachtenden Nachbarn klar. Auf diese Art läuft das Programm, vom Einstiegspunkt ausgehend, solange durch das Gewässernetz, bis es dort wieder ankommt und sammelt alle unterwegs gefundenen Abschnitte auf der Ringstruktur ein. Die geordnete Bewegung innerhalb der Kreisstruktur ordnet auch die Linien und ihre Ausrichtung im resultierenden Array, aus dem im Abschluss des Rücklaufes ein Polygon erzeugt werden kann. Die hier beschriebenen Vorgänge sind durch **Code 6** im Programm

[16] ARC/INFO beispielsweise bietet dank seiner Topologie die Möglichkeit, das hier dargestellte Problem sehr einfach zu lösen, indem das Linien- in ein Polygoncover konvertiert wird; die resultierenden Polygone sind dort die Ringstrukturen. ARCVIEW und auch MAPOBJEKTS haben dafür keinen Befehl. Hier muss also eine andere Strategie verfolgt werden.

umgesetzt[17]. Diese Subroutine kommt innerhalb der Extension bei verschiedenen Arbeitsschritten zum Einsatz. Deren Beschreibung beruft sich ggf. auf **Code 6**.

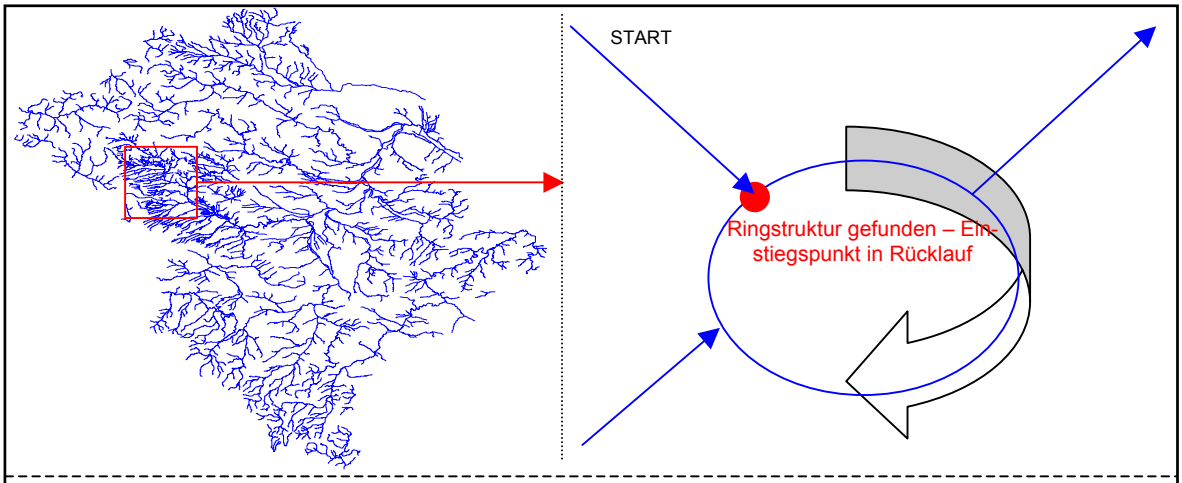


Abb. 32: Ablauf innerhalb des Algorithmus beim Auffinden von Ringstrukturen (UNSTRUT)
Rechts: wird der Einstiegspunkt wiedergefunden, so liegt eine Ringstruktur vor

Die gefundenen Ringstrukturen werden in der Tabelle markiert und sind somit als separates Linien- oder Polygonthema speicherbar. Der Bearbeiter hat dadurch einerseits die Möglichkeit, Standgewässer zu extrahieren, andererseits kann er nun diejenigen Abschnitte entweder physisch entfernen oder durch Attribute gesondert ausweisen, die nicht in das Gewässernetz einbezogen werden sollen.

Dieses System funktionierte auch bei Braided-River-Strukturen, in denen das verwilderte Flussnetz im Spannungsfeld zwischen Reliefenergie und Akkumulation viele benachbarte „Ringe“ erzeugen kann (vgl. **Abb. 33**). Dafür werden die Kreisstrukturen zugeordneten Abschnitte innerhalb des Algorithmus als solche markiert, um bei der Bearbeitung des benachbarten Rings gesondert behandelt werden zu können.

Abb. 33: Braided River System (SAGAVANIRKTOK RIVER, Alaska, Quelle: [RITTER ET AL 1995])

[17] Der dort dunkelgrau mit weißem Text hervorgehobene Teil betrifft diese Problemlösung speziell.

4.1.3.8 Analyse des Gewässerzusammenhangs - Clusterbildung

Unter einem Cluster soll in dieser Arbeit ein Komplex zusammenhängender Linien verstanden werden. Ist er baumstrukturiert, kann man ihn auch als Teilbaum im Sinne der Graphentheorie (vgl. **Kap. 4.2.5.1**) auffassen. Die Abschnitte eines Clusters erhalten in einem Attribut (z. B. `Cluster`) denselben Wert. Mithilfe solcher Cluster kann beispielsweise festgestellt werden, ob das Gewässernetz noch nicht entdeckte Lücken oder Einzellinien enthält.

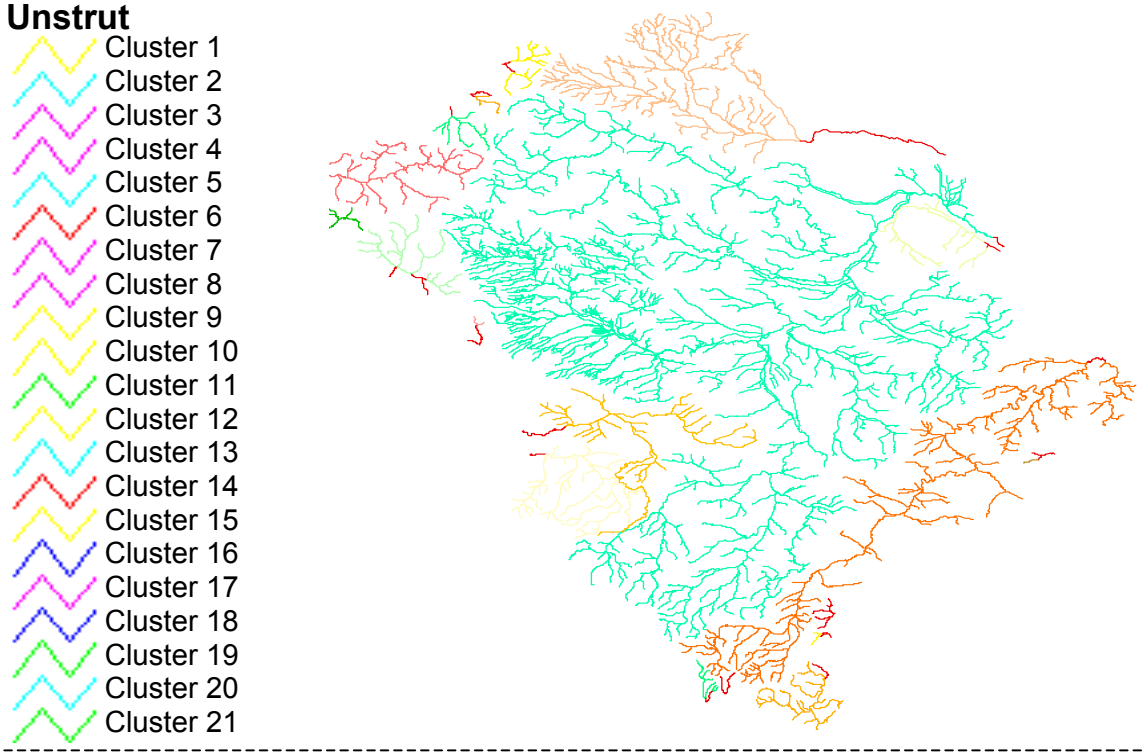


Abb. 34: Analyse des Gewässernetz-Zusammenhangs (UNSTRUT)
Die Legende und die unterschiedlichen Farben geben einen Überblick zur Aufteilung des Gesamtgewässernetzes in verschiedene Gewässerbäume. Der Bearbeiter entscheidet auf dieser Basis, ob und welche Einzelnetze zu einem Gesamtnetz zusammengeführt werden sollen.

Ein Cluster kann demnach ein einziger Abschnitt aber auch das Gewässernetz z. B. der SAALE oder in diesem das der UNSTRUT sein, sofern letzteres geometrisch oder attributiv nicht an die SAALE angeschlossen ist. Mithilfe dieser Funktion kann man also sehr übersichtlich den Zusammenhalt innerhalb des Gewässernetzes darstellen und überprüfen (**Abb. 32**). Der Algorithmus ist an **Code 6** angelehnt. Die Startlinie wird vom Programm selbst gesucht (vgl. **Kap. 4.1.3.10**). Bei Vergabe eines Höhenattributes wird der jeweilige Gebietsauslass gefunden. Ist ein Cluster durchlaufen und sind damit nicht alle Datensätze des Gewässernetzes eingefangen, werden die übrigen Linien solange zusammengefasst, bis alle Abschnitte einem Cluster zugeordnet wurden. Die Klassifizierung nach dem räumlichen Zusammenhang erleichtert auch die weitere Bearbeitung des Gewässernetzes. So lassen sich beispielsweise auf Basis des hier belegten Attributes `Cluster` (siehe Legende in **Abb. 34**) zusammenhängende Strukturen separat selektieren und als eigenständiges Thema exportieren und weiterreichen. Auch die unter **Kap. 4.2** besprochene Hierarchisierung profitiert davon.

4.1.3.9 Integration von Standgewässern oder beidseitig digitalisierten Flüssen

Seen und Rückhaltebecken sowie beidseitig digitalisierte Flüsse stellen in den meisten Flussnetzsystemen wahrscheinlich die am häufigsten auftretenden Kreisstrukturen dar. In der Regel werden diese Objekte auch in einem separaten Layer festgehalten. Um solche Elemente weitgehend automatisiert in das Netzwerk einzubinden, muss ein Weg durch das Gewässer selbst gefunden werden, da der natürliche Verlauf des fließenden Wassers für gewöhnlich nicht dem Umriss folgt. Dabei erscheint der Ansatz, den Flusslauf durch hydrologische Modelle zu simulieren, zumindest für die gegebene Fragestellung unnötig aufwendig. Als Vorschlag für alle möglichen Formen eines Sees und seines Strömungsverhaltens wurde eine Lösung angestrebt, bei der die Verbindung der einzelnen Umriss-Knoten zum Auslauf-Knoten auf Basis der Konstruktion künstlicher Verbindungslinien abgeschätzt wird [LEHNER 1998]. Diese sollen unter allen erdenklichen Umständen stets etwa in der Mitte der Kreisstruktur zwischen rechtem und linkem Ufer verlaufen.

Unter verschiedenen Ansätzen wurde als der stabilste die Erzeugung von Thiessen-Polygonen ausgewählt. Dies sind *„... polygons bounding the region closer to a point than to any adjacent point. The polygons are drawn so that the lines are of equal distance between two adjacent points"* [BAILEY & GATRELL 1995, S. 45]. Thiessen-Polygone, auch als VORONOI-Diagramme oder DIRICHLET-Tessellationen bekannt, stellen eine grobe Form der Interpolation – vor allem in GIS – dar [BARTHELME 1995]. In der Entwicklungsphase wurden dazu möglichst unterschiedliche und komplizierte Formen von Seen digitalisiert (**Abb. 35**). Das Verfahren wurde anschließend an Ringstrukturen innerhalb der Testgebiete (v. a. TANGER) erprobt.

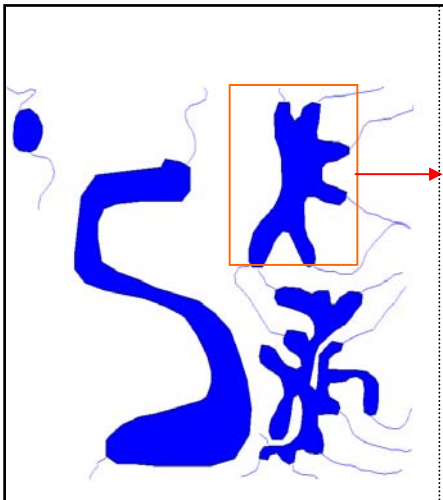


Abb. 35: Testdatensatz unterschiedlicher, komplizierter Seen / Ringstrukturen

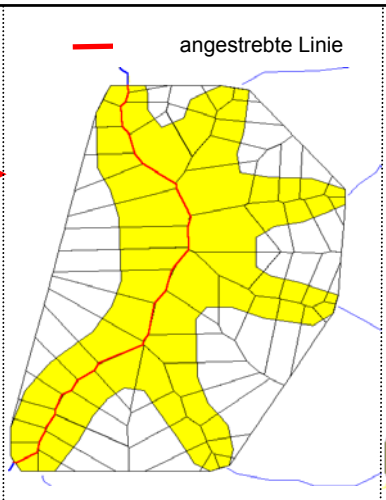


Abb. 36: Polygonnetz

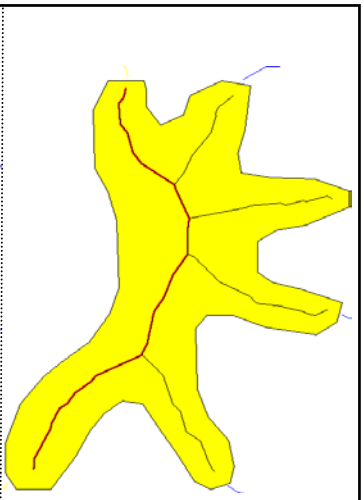


Abb. 37: Reduziertes Polygonnetz

Da sich in einem Netzwerk von Thiessen-Polygonen ihre Ränder immer genau zwischen zwei benachbarten Punkten befinden (**Abb. 36**), liegt der Gedanke nahe, diese Eigenschaft zur Bestimmung einer geschätzten Mittellinie durch einen beliebig geformten Seeumriss zu nutzen, indem als Basis des Polygonnetzes die Stützpunkte der Umrisslinie des Sees herangezogen werden (**Abb. 36, 38** und **39**). Aus **Abb. 38** und **Abb. 39** ist ersichtlich, dass die Genauigkeit der simulierten Mittellinie von der Anzahl auf dem Umriss befindlicher Punkte abhängt. Diese kann durch den Benutzer modifiziert werden. Allerdings

wachsen die Rechenzeiten mit zunehmender Dichte von Stützpunkten[18]. Die zugrundeliegenden Algorithmen wurden der Fragestellung angepasst, d. h. alle rechenintensiven Vorgänge wie Pufferung und Verschneidung auf das absolut notwendige Maß reduziert. Letztendlich gilt es, wie **Abb. 40** verdeutlicht, eine Reihe von Linien zu erzeugen, aus denen die gewünschte Mittellinie mit möglichst wenig Aufwand zu extrahieren ist.

Dieser Aufwand besteht darin, alle hier gefundenen Segmente auf den See-Auslauf hin zu orientieren und im nächsten Schritt vom betrachteten Zufluss (A) zum Ablauf (B) (**Abb. 36, 37** und **40**) hin zu „routen". Die dabei gefundenen Abschnitte repräsentieren die gesuchte Mittellinie. Alternativ kann jedoch auch das gesamte „Gewässernetz" des Sees aufgebaut und die Orientierung nach dem bereits in **Kap. 4.1.3.5** beschriebenen Prinzip der Gewässerausrichtung vorgenommen werden (**Abb. 41**).

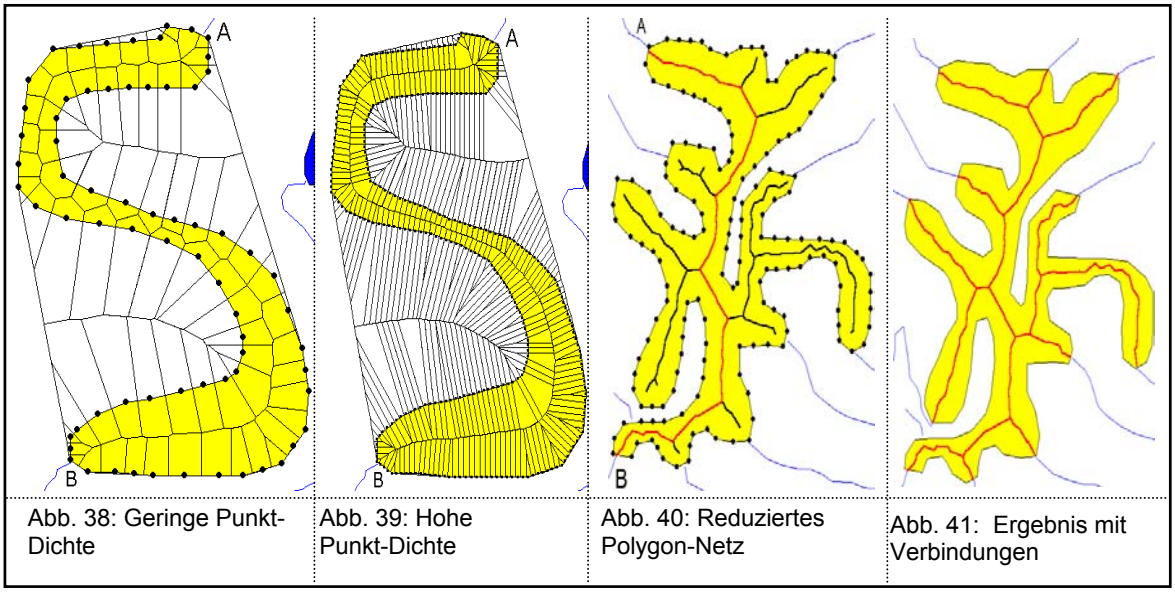


Abb. 38: Geringe Punkt-Dichte

Abb. 39: Hohe Punkt-Dichte

Abb. 40: Reduziertes Polygon-Netz

Abb. 41: Ergebnis mit Verbindungen

4.1.3.10 Ermittlung des Gebietsauslasses

Besonders in verteilten Gewässernetzen (z. B. das Abflusssystem Dänemarks) kann eine Funktion zur automatischen Erkennung des jeweiligen Gebietsauslasses sehr sinnvoll sein. Leider stehen dafür nicht immer ausreichend genaue Höhendaten zur Verfügung – problematisch besonders in Gebieten mit geringer Reliefenergie. J. PAIVA stellte 1998 einen Algorithmus zur Erfassung des Gebietsauslasses vor, der auf neun heuristischen Fallunterscheidungen basiert [DE SERRES & ROY 1990], die in **Tab. 5** dargestellt sind und ihre Grundlagen in der Gewässermorphologie haben.

FALL	WINKEL C1,C3	WINKEL C1,C2	WINKEL C2,C3	UNTERLIEGER
C 1 C 2 C 3	= 180°	= 90°	= 90°	C1 oder C3
C 1 C 2 C 3	= 180°	< 90°	> 90°	C3

[18] Viele GIS stellen diese Funktion standardmäßig zur Verfügung. Innerhalb von ARCVIEW ist dies jedoch nicht der Fall. Grundlage der Implementierung ist eine Skript von ESRI [www.arcscripts.esri.com] zur Erzeugung von Thiessen-Polygonen.

C 1 C 3 C 2	= 180°	> 90°	< 90°	C1
C 1 C 2 C 3	< 180°	<= 90°	>= 90°	C3
C 1 C 3 C 2	< 180°	>= 90°	<= 90°	C1
C 1 C 2 C 3	> 180°	<= 90°	>= 90°	C3
C 1 C 2 C 3	> 180°	>= 90°	<= 90°	C1
C1 C2 C3	> 180°	>= 90°	>= 90°	C3 wenn 1,2 < 2,3 C1 wenn 2,3 < 1,2
C1 C2 C3	< 180°	< 90°	< 90°	C3 wenn 1,2 < 2,3 C1 wenn 2,3 < 1,2

Tab. 5: Fallunterscheidung zu den Anschlusswinkeln in Kreuzungsbereichen zur Ermittlung des Gebietsauslasses (nach [DE SERRES & ROY 1990])

Der Prozess der Ausrichtung umfasst vier Schritte:

- Bestimmung der Fließrichtung auf Basis der Annahmen aus **Tab. 5**
- Bestimmung des Unterliegers für jeden Abschnitt (**Abb. 42**)
- Identifizierung der Äste des Netzwerkes
- Bestimmung des Gebietsaulasses (**Abb. 43**).

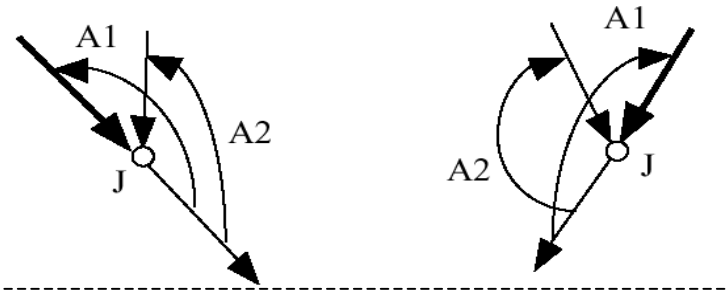


Abb. 42: Unterlieger (Parent Chanels) an Kreuzungen (nach [PAIVA & EGENHOFER 1997])
links: entgegen Uhrzeigersinn
rechts: mit dem Uhrzeigersinn

Dieses Verfahren wurde innerhalb der Arbeit so modifiziert, dass der vorhandene Algorithmus zur Erfassung der Außenpunkte (vgl. **Kap. 4.1.2**, **Code 2**) genutzt wird, um von jedem dieser Punkte ausgehend mithilfe der Winkelregeln in **Tab. 5** in das Gewässernetz „hineinzulaufen“, bis entweder eine auf diesem Weg gefundene Kreuzung zeigt, dass die Bewegungsrichtung flussabwärts erfolgt und der Ausgangspunkt somit als Quelle identifiziert wurde, oder ein Außenpunkt, bei dem auf diesem Weg nur winkelbestimmte Zu- und lokale Hauptflüsse

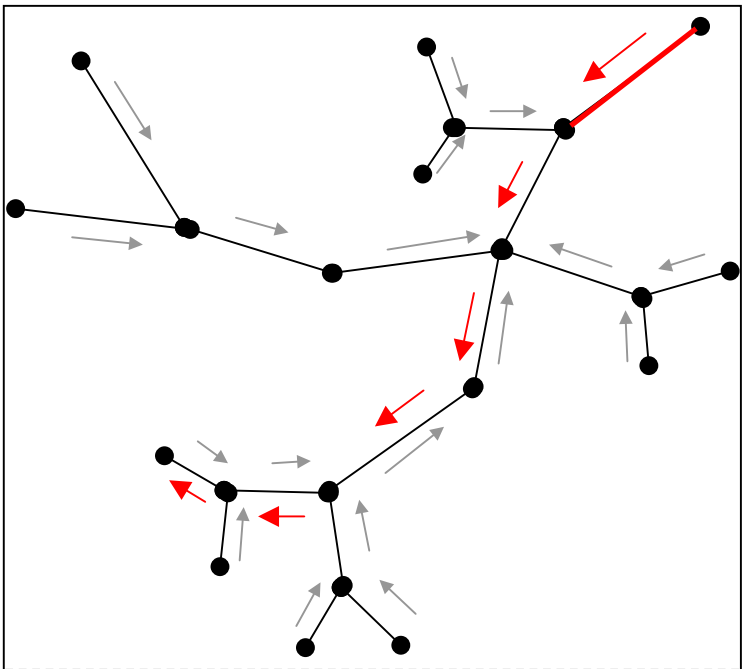


Abb. 43: Schematische Darstellung zur Ermittlung des Gebietsauslasses (Modifikation nach [PAIVA & EGENHOFER 1997])
- nur die roten Pfeile begegnen keinem winkelbasierten Ausschlusskriterium und gehen somit vom Gebietsauslass aus

angetroffen wurden, als Gebietsauslass identifiziert und in einem entsprechenden Attribut als solcher festgehalten werden kann (**Abb. 43**, rote Pfeile). Die Ermittlung der Winkelbeziehungen erfolgt intern mit einer Routine, die der in **Kap. 4.2.5.2** Beschriebenen ähnelt (vgl. auch **Abb. 42**).

Für die getesteten Datenbasen erwies sich der Algorithmus als stabil und effizient. Die Untersuchungen von DE SERRES und ROY [DE SERRES & ROY 1990] sowie von J. A. PAIVA und M. EGENHOFER [PAIVA & EGENHOFER 1997] haben gezeigt, dass diese Winkelregeln (**Tab. 5**) mit 88% Wahrscheinlichkeit zu einem korrekt ausgerichteten Gewässernetz und dem wirklichen Gebietsauslass führen. Obwohl die Ergebnisse innerhalb dieser Arbeit auf eine noch höhere Zuverlässigkeit hindeuten, sollte das Ergebnis immer vom Bearbeiter überprüft werden.

4.1.4 Zusammenfassung

Die unter **Kap. 4.1** beschriebenen Funktionen einer ARCVIEW-Erweiterung wurden auf Grundlage der Anforderungen an ein zu hierarchisierendes Gewässernetz gemeinsam mit verschiedenen Institutionen und Firmen an unterschiedlichen Datengrundlagen (vgl. **Kap. 1.2**) entwickelt. Dabei flossen über den gesamten Entwicklungszeitraum durch die unterschiedliche Perspektive und Ausgangsbasis der einzelnen Bearbeiter immer wieder neue Probleme, Hinweise und Verbesserungsvorschläge in den Entwicklungsprozess ein. Naturgemäß können die Ergebnisse nur ein Kompromiss aus diesen Einflüssen und den Prämissen der eigenen Zielvorgaben, aus Übersichtlichkeit und Funktionsumfang sowie eine Arbeitsvorlage zum hierarchieorientierten Präprozessing digitaler Gewässernetze sein und haben sich als solche auch immer wieder bewährt.

Das System ist dabei jederzeit ergänzbar, aktualisierbar und anpassbar. Die Automatisierung der beschriebenen Vorgänge ermöglicht deren Einbindung in den laufenden Bearbeitungsprozess von Gewässernetzen. So kann die Applikation auf Veränderungen der Datenbasis „selbständig“ reagieren. Werden beispielsweise neue Knoten (z. B. eine neu installierte Messstelle) in die Geometrie eingefügt, kann die Analyse und Vorbereinigung des topologischen Netzwerkes im Hintergrund erfolgen und so die Ergonomie und Effektivität beim Umgang mit diesen Daten wesentlich erhöhen.

An diese Vorbereitungen schließt sich die eigentliche Hierarchisierung des geometrisch bereinigten Gewässernetzes an (vgl. **Abb. 1**), wie sie in den folgenden Kapiteln beschrieben werden soll.

4.2 Abbildung von Hierarchien in relationalen Datenbanksystemen

4.2.1 Einführung

> *" There is more than one way to do it" (Larry Wall, Entwickler der Programmiersprache Perl)* [ACM 2002].

Hierarchien können, sofern sie dem DBMS (Datenbankmanagementsystem) bekannt sind, für viele Anwendungen, wie in der Medizininformatik bei der Modellierung des Blutgefässsystems des menschlichen Körpers (z. B. [SMITH ET AL. 2002]), die Korrespondenzstruktur einer Internetseite oder die räumlichen Zusammenhänge innerhalb eines Fließgewässereinzugsgebietes genutzt werden. Die meisten modernen DBMS unterstützen Hierarchien jedoch nicht explizit[19]. In diesen Fällen sind Metadaten zur Beschreibung der hierarchischen Zusammenhänge notwendig. Die hier dargestellten Ansätze, Hierarchien in ein Datenbanksystem einzuführen, beruhen darauf, relationale Datenbanken durch Schemainformationen um Hierarchien zu erweitern.

Dabei wurde das Ziel verfolgt, den Standard-SQL-Umfang und die Grundstruktur des Datenbankkonzeptes (vgl. **Kap. 2.4.1**) für diese Zwecke auszunutzen, ohne auf spezielle, DBMS-abhängige Erweiterungen angewiesen zu sein. Es soll gezeigt werden, wie mithilfe von Metadaten (Attribute, Tupel) über Hierarchien semantische Query-Optimierungen und verbesserte Datenspeicherung im Fließgewässermanagement erreicht werden können (**Kap. 4.3.1**).

Ein vektorbasiertes GIS ermöglicht die Belegung jedes räumlichen Objektes mit einer eindeutigen Identifikationsnummer [SAURER & BEHR 1997]. Für die Hierarchisierung bedeutet dies, dass die individuellen Flussabschnitte erst zu einem solchen System zusammengeführt werden müssen. Dabei sollen die topologischen Beziehungen und geeignete Attribute das Fließen des Wassers reflektieren. Wo immer sich Wasser mit seinen Inhalten an einem Abschnitt befindet, muss es von dort aus entlang dem Gewässernetz zum Gebietsauslass fließen - dem Ausgangspunkt der Hierarchisierung und der Lösung damit verbundener Fragestellungen (vgl. **Kap. 1.2**, **Abb. 43**).

Zunächst soll ein Überblick über die untersuchten und ggf. implementierten Formen der Modellierung von Hierarchien in relationalen Datenbanksystemen gegeben werden. Darüber hinaus wurden eigene, der Zielstellung der Arbeit entsprechende Ansätze entwickelt, getestet und, sofern sie in diesem Sinne eine Verbesserung darstellen, beschrieben und in Applikationen integriert.

4.2.2 Unter-Oberlieger-Beziehungen („Modell angrenzender Listen")

4.2.2.1 Einführung

Hierarchische Daten sind Informationen in Tabellen, die über ein oder mehrere Attribute die Relationen eines bestimmten Informationsbereiches widerspiegeln. Dieses Kapitel soll sich mit den Möglichkeiten der Erfassung, Abbildung und Nutzung von Unter-Oberlieger-Beziehungen beschäftigen. Dabei werden, wie der Name schon sagt, zwei Tupel benötigt: eines für die ID jedes Abschnittes und ein anderes zur Identifizierung seines flussabwärtigen Nachbarn. Das System ist sehr speichereffizient und intuitiv. Darüber hinaus stellt es

[19] abgesehen von hierarchischen Datenbankmanagementsystemen, deren Bedeutung in den vergangenen Jahren jedoch stark zurückgegangen ist.

international in vielen Institutionen einen Quasi-Standard dar, beispielsweise bei hydrologischen Modellierungen [JENSON & DOMINGUE 1988, BECKER ET AL. 2000].

Hierarchiebezogene SQL-Statements setzen ein normal-hierarchisches Gewässersystem (vgl. **Kap. 2.3**) voraus. Innerhalb der Arbeit wurden dabei folgende Methoden untersucht und zum Teil entwickelt:

- Rekursion
- Stack
- eindimensionale Tabelle
- rekursive Joins
- indizierte Arrays.

Dabei sah das Datenmodell wie folgt aus:

Feld	Typ	Kommentare
Name	Varchar(x)	Name des Abschnittes
INFO		Synonym für alle abzufragenden Felder
FGWID	Numeric	Die einmalige ID für jeden Datensatz
FGWULID	Numeric	Die FGWID des unterliegende Objektes (Parent-ID)

Tab. 6: Datenmodell des Unter-Oberlieger-Tests

Name	FGWID	FGWULID	Hiedist
Elbe	1	-1	0
Saale	13	1	1
Salza	45	13	2
Querne /Weida	1424	45	3
Mulde	23	1	1
Elster	167	23	2

Tab. 7: Beispiel zu einer Tabelle des Unter-Oberlieger-Tests. Das Attribut **Hiedist** spiegelt die hierarchische Entfernung des Abschnittes von der Quelle wider.

Tab. 7 beinhaltet die hierarchische Information in den Feldern FGWID und FGWULID[20]. FGWID ist die eindeutige Identifikationsnummer des Abschnittes (Fließgewässer-ID), FGWULID die seines Unterliegers (Fließgewässer-Unterlieger-ID). Sie attributiert das "Modell angrenzender Listen" nach der gleichnamigen Technik aus der Graphentheorie (vgl. **Kap. 4.2.5.1**). Die Paare der Abschnitte (bzw. Kanten in der Terminologie dieser Theorie) grenzen aneinander. Dabei wird der Unterlieger entsprechend seiner Position in der Baumstruktur als Vater und alle diejenigen Abschnitte, de-

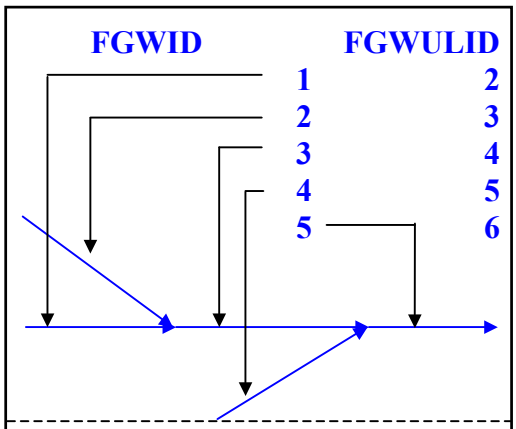


Abb. 44: Unter-Oberliegerbeziehungen

[20] Die Einrückung in der Spalte NAME soll die hierarchischen Zusammenhänge unterstreichen.

ren FGWULID dessen FGWID entspricht, als seine Kinder bezeichnet. **Abb. 44** schematisiert diesen Zusammenhang.

In der elektronischen Datenverarbeitung spielt die Liste (Array) wegen ihrer Flexibilität eine große Rolle. Listen verknüpfen eine nichtleere Menge von Datenelementen, von denen jedes Element mit Ausnahme des letzten einen Nachfolger und, mit Ausnahme des ersten, einen Vorgänger hat. Nach dieser Definition hat jedes der einzelnen Elemente eine feste Position innerhalb dieser Kette. Damit besteht zwischen den Elementen eine Ordnungsrelation [ZIETLOW 1993]. Auf Listen werden normalerweise folgende grundlegende Operationen angewandt:

- füge Elemente ein (u. U. nach bestimmten Kriterien)
- suche Elemente
- entferne Elemente

Ausgehend von der oben genannten Definition einer Liste sind zahlreiche Implementierungen denkbar. Die naheliegendste ist die einfach verkettete Liste (engl. single-linked List), d.h. jedem Datenelement wird ein Zeiger auf die Adresse des Nachfolgers mitgegeben, der im Fall des letzten Elements auch leer sein kann, da dieses keinen Nachfolger hat. Verallgemeinert man die Definition einer Liste und weist dem letzten Element die Adresse des ersten zu, so entsteht eine Ringstruktur. Diese Ringstrukturen, wie einfache Listen auch, können mehrfach ineinander verschachtelt sein, wodurch wiederum komplexere Strukturen (Vielfachringe, z. B. Braided-River) entstehen. Sie sind in [FINDEISEN 1990] anschaulich beschrieben.

Bei der Zuweisung der FGWULID zu den FGWIDs wird zunächst allen Abschnitten eine eindeutige ID (FGWID) zugewiesen. Für jeden Abschnitt gibt es laut Definition (vgl. **Kap. 2.3**, **Kap. 3.1.1**) genau einen Unterlieger. Dieser beginnt dort, wo der gerade untersuchte Abschnitt in seiner ausgerichteten Version endet. Ist also der Unterlieger gefunden, kann aus der Attributtabelle seine FGWID gelesen und dem betrachteten Abschnitt als FGWULID zugewiesen werden (vgl. **Code 7**).

Die folgenden Untersuchungen konzentrierten sich darauf, sowohl im GIS als auch in der Datenbank hierarchische Abfragen zu ermöglichen und diese weitestgehend effektiv und dabei allgemein anwendbar zu gestalten. Es sollte eine möglichst optimale Eigenschaftskonstellation gemäß **Tab. 2** und **3** bei der Anwendung dieses Modells gefunden werden.

4.2.2.2 Rekursion

In einigen Veröffentlichungen (z. B.[SHEKHAR 1993], [TIMPF & FRANK 2000]) wird die Rekursion als die eleganteste Methode beschrieben, um hierarchische Daten zu durchlaufen, da nur eine Funktion benötigt wird, um alle Daten zu prozessieren, indem ein Unterprogramm sich selbst immer wieder aufruft. Die in **Code 8** dargestellte Funktion wird einmal initiiert (mit der Wurzel des Baumes – dem gewählten Gebietsauslass) und bewegt sich in die Äste hinein bis zu den Kindern ohne folgende Generation, den Quellen. Sie kann von einem beliebigen Subnode aus den dazugehörigen Subtree (Unterbaum) finden. Es ergeben sich jedoch folgende Nachteile, welche die Rekursion für diese Arbeit ungeeignet machen:

- Die meisten Systeme unterstützen Rekursion nur eingeschränkt. ARCVIEW beispielsweise erlaubt keine Rekursion.

- Es ergab sich ein logarithmischer Abstieg der Geschwindigkeit mit zunehmender Hierarchietiefe unter VISUAL BASIC und MSACCESS[21] (**Tab. 8**).

Durchlauf n	Milisekunden
100	3*n
200	29*n
300	53*n
400	79*n
500	103*n

Tab. 8: Geschätzte Ausführungsgeschwindigkeit der Rekursion

- Es ist nicht möglich, eine solche Rekursion mit dem SQL-Standard umzusetzen. Dazu sind Stored Procedures[22] nötig, wodurch sich vor allem folgende Nachteile ergeben:
  - Möglicherweise lassen sich Stored Procedures vom zugrundeliegenden Datenbank-System nicht nutzen oder man hat nicht den entsprechenden Zugang, um sie anzulegen.
  - Stored Procedures lassen sich in der Regel nicht von GIS nutzen.

4.2.2.3 Stack

Die Arbeit mit einem Stack[23] ist eine andere Methode zum Management großer Listen. Bei der Arbeit mit hierarchischen Listen empfiehlt sich das FIFO („First In First Out")-Modell. Dabei wird ein Stack erzeugt und diesem die FGWID desjenigen Datensatzes hinzugefügt, mit dem gerade gearbeitet wird. Nach der Auswertung dieses Datensatzes kann er aus dem Stack entfernt werden (pull). Hat der Abschnitt Kinder, so wird mit dem ersten Kind weitergearbeitet und dessen FGWID an das Ende des Stacks geschickt (push), solange, bis alle Datensätze durchlaufen sind. Die Vorteile dieser Methode liegen vor allem in der Nähe zur Middleware[24] – also dem Outputbereich des Programms:

- Will man die Ergebnisse beispielsweise in HTML ausgeben, so kann das nach jeder Iteration geschehen
- Die Ausführung erfolgt näher zu den Daten und damit schneller
- Es wird nur eine Verbindung zur Datenbank benötigt.
- Ähnlich wie bei den DB-Verbindungen, die im Rekursiv-Modell erstellt wurden, ergeben sich jedoch folgende Nachteile:
- Es kann schwierig sein, den Code der Stored Procedures zu verstehen. **Code 9** zeigt das Beispiel eines hierarchischen Stack-Algorithmus aus der Microsoft SQL 6.5 Dokumentation.
- Sie entsprechen nicht dem SQL-Standard .

[21] Diese Geschwindigkeitsmessungen wurden mit einer dafür programmierten VISUAL BASIC - Anwendung durchgeführt.
[22] Stored Procedures (gespeicherte Befehlsabfolgen) sind Sammlungen von SQL-Statements. Bei der ersten Ausführung einer Stored Procedure wird im DBMS ein Ausführungsplan erstellt, der die nächsten Ausführungen der Befehlsfolge gegenüber nichtgespeichertem SQL wesentlich beschleunigt. Andererseits erlauben sie auch SQL-fremde und DBMS-abhängige Konstrukte wie z. B. Schleifen [SCHULZE 2000].
[23] Stack (Keller): „*Ein Verfahren, bei dem Daten geordnet in einen Speicher eingegeben und bei Bedarf wieder heraus geholt werden können. Dabei wird der zuerst „eingekellerte" Wert als erster wieder „ausgekellert" (First In Last Out - FILO) bzw. umgekehrt (First In First Out – FIFO). Wie viele Objekte in einem Stack gespeichert werden können, hängt von seiner Tiefe ab, die von Fall zu Fall anders definiert werden kann*" [SCHULZE 2000, S. 203].
[24] Software, die Anwendungen zum Datenaustausch verbindet [SCHULZE 2000] (vgl. Kap. 3.3.2).

Mit diesen Nachteilen werden auch Stacks der Aufgabestellung nicht gerecht.

4.2.2.4 Eindimensionale Tabelle („Flat Table Model“)

Eindimensionale (flache) Tabellen beinhalten alle hierarchischen Metadaten, so dass sie mit relativ einfachen Abfrage-Funktionen ausgewertet werden können (**Tab. 9**). Die hierarchische Distanz im Attribut HIEDIST spiegelt sich im Einrücken der Gewässerbezeichnung in der linken Spalte von **Tab. 7** wider. Das Datenschema kann demnach z. B. wie folgt definiert werden:

Feld	Typ	Kommentare
FGWID	Numeric	die einmalige ID für jeden Datensatz
Name	Varchar(x)	Name des Abschnittes
FGWULID	Numeric	die FGWID des unterliegende Objektes (Parent-ID)
Info		Synonym für alle abzufragenden Felder
Position	Numeric	Darstellungsreihenfolge
Hiedist	Numeric	Hierarchische Entfernung (Anzahl der Zusammenflüsse) zwischen Abschnitt und Gebietsauslass

Tab. 9: Datenschema einer eindimensionalen (flachen) Tabelle im Unter- Oberliegermodell

Die Datenselektion erfolgt beispielsweise mit der folgenden Funktion (Pseudocode):

CODE 10: PSEUDOCODE ZUR DATENSELEKTION AUF BASIS ANGRENZENDER LISTEN IN EINER FLACHEN TABELLEN

```
Function DisplayChildren()
  SELECT FGWID, INFO, HIEDIST 'SQL-Statement
  FROM river
  ORDER BY POSITION

  while das Recordset nicht leer ist {
      (indent by Hiedist)
  }
End Function
```

Die Schwierigkeit besteht vor allem in der Zuweisung der Darstellungsreihenfolge. Die Position eines neuen Kindes (Child) sollte größer sein, als sein Parent (Elternteil), jedoch kleiner als irgendeins seiner Kinder oder der nächsten Geschwister (Siblings) auf demselben Level. Im Beispiel von **Tab. 9** (Erweiterung der **Tab. 6**) sollte die SAALE eine Position 2 haben. Da jedoch alle Einträge in der Tabelle von 1 bis 7 positioniert wurden, bedeutete dies, dass die Darstellungsreihenfolge aller Gewässer mit einer Position größer 1 inkrementiert werden muss.

Die Performance des Rekursions- und des Stackmodells lassen sich nach den vorgenommenen Messungen im Mittel folgendermaßen beschreiben $(n * 2) + (n * 2* x) = x*4n$ [25]. Für das Flat Table Model gilt angenähert diese Regel: $(n * 2) + (n * 2 * x) + (n * 2) = x*6n$[26].

Neben dem Verlust durch die Einrückung spricht die Notwendigkeit von Stored Procedures gegen diesen Ansatz. Auch diese Methode ist aus den schon in **Kap. 4.2.2.2** und

[25] Verbindung + Einrückung; n ist die Dauer des jeweilige Datenbankzugriffs
[26] Verbindung + (Aktualisierung * Datensätze(x)) + Einrückung; n = Dauer des jeweilige Datenbankzugriffs, x = Anzahl der Datensätze

Kap. 4.2.2.3 beschriebenen Gründen kaum einem breiten Kreis von Anwendern zugänglich zu machen und somit weder generell anwendbar noch übertragbar.

4.2.2.5 Rekursive Joins

Ein Join selektiert Werte aus mehreren Tabellen über ein oder mehrere Schlüsselfelder in jeder der beteiligten Tabellen.

Ein rekursiver Join geht hingegen immer wieder von den Werten einer Spalte in einer Tabelle auf eine Referenz-Spalte in derselben Tabelle zurück. Grundlage rekursiver Joins sind hierarchische Metadaten. Eine hierarchische Tabelle beinhaltet also eine Spalte, die eine Referenz auf eine andere Spalte in derselben Tabelle hat. Dementsprechend könnte man auch bei diesem Ansatz von einer flachen Tabelle sprechen.

- Ein an rekursive Joins gekoppeltes hierarchisches `SELECT`-Statement kann z. B. im Bereich des Personalmanagements für folgende Fragen genutzt werden (**Abb. 45** und **Abb. 46**):
- Welche Personen sind einem Angestellten direkt oder indirekt unterstellt? Da sich diese Art der Fragestellung den Hierarchie–Baum abwärts bewegt, wird sie als abwärtsgerichtet bezeichnet[27].
- Welche Reihe von Personen führt direkt von einem beliebigen Angestellten zum Betriebsleiter? Da sich diese Art der Fragestellung den Hierarchiebaum aufwärts bewegt, wird sie als aufwärtsgerichtet bezeichnet.

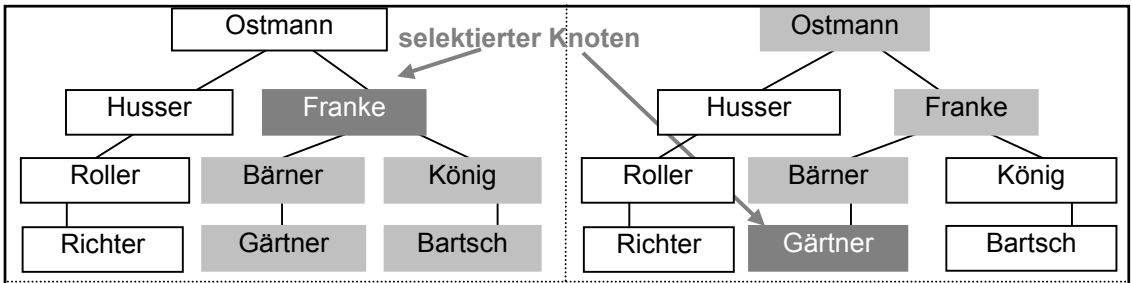


Abb. 45: Abwärts gerichtete Query mit Subtree

Abb. 46: Aufwärts gerichtete Query

Wenn eine Tabelle hierarchische Daten enthält, ist es demnach möglich, eine Untermenge dieser Daten mithilfe einer hierarchischen Abfrage herauszufiltern. Die Untersuchungen zu rekursiven Joins beschränkten sich auf das Datenbanksystem ORACLE. Der Ansatz wurde jedoch auch von IBM aufgenommen und erscheint in abgewandelter Form im SQL-Server von Microsoft und sicher auch in anderen DBMS. ORACLE ermöglicht dies durch eine Kombination verschiedener SQL-Befehle, welche die gesamten Baumstruktur der Daten einer Tabelle in einer Abfrage zurück geben können:

- `START WITH`: hier wird der Start-Eintrag (z. B. die ID des Einzugsgebiets-Auslaufes, die Wurzel) angegeben.
- `CONNECT BY PRIOR`: hier wird die Beziehung zwischen über- und untergeordneten Datensätzen (Eltern- und Kindreihen) angegeben.
- `WHERE`: Schränkt die zurückgegebenen Datensätze weiter ein.

[27] Wie so oft in der Informatik wird die erfahrene Welt auch hier auf den Kopf gestellt: selbst Bäume stehen mit der Wurzel nach oben und den Blättern nach unten. Auch der Bezeichnung Unterlieger wird damit widersprochen. Da sich diese Auffassung von Hierarchien jedoch etabliert hat, soll sie auch in dieser Arbeit vertreten werden.

Diese Unterstützung von Hierarchien ist jedoch recht begrenzt und wenig intuitiv. ORACLE benutzt die Informationen in dieser SQL-Abfrage, um die Baumstruktur durch die im folgenden beschrieben Schritte innerhalb eines rekursiven Joins abzubilden:

- Diejenigen Wurzel-Datensätze werden selektiert, welche die Bedingung der `START WITH`-Klausel erfüllen.
- Das DBMS selektiert die nächste Hierarchieebene (Kindreihen). Jeder untergeordnete Datensatz muss die Bedingung der `CONNECT BY`-Klausel erfüllen – also mit einer der Wurzeln verbunden sein.
- Anschließend werden sukzessive (rekursiv) die Generationen von Kindreihen herausgearbeitet. ORACLE selektiert zuerst die Kinder der aus Schritt 2 resultierenden Datensätze, danach die Kinder dieses Ergebnisses und so weiter, indem die `CONNECT BY`-Bedingung in Bezug zur gegenwärtigen Eltern-Reihe geprüft wird.
- `PRIOR Feld1 = Feld2` weist das DBMS an, die Zeile so darzustellen, dass die Werte von Feld1 immer gleich denen von Feld2 sind. Hierdurch kann also bestimmt werden, ob die Abfrage aufwärts oder abwärts gerichtet ist.
- Wenn die Abfrage eine `WHERE`-Klausel enthält, entfernt ORACLE alle Datensätze aus der Rückgabe, die der Einschränkung durch `WHERE` nicht entsprechen.

Dabei unterliegt ein `SELECT`-Statement innerhalb einer hierarchischen Abfrage folgenden Einschränkungen:

- Wenn eine `ORDER BY`-Klausel in der hierarchischen Abfrage verwendet wird, ordnet das DBMS die Datensätze nach `ORDER BY` und nicht wie unter Schritt 5 beschrieben [LOCKMAN 1998].

Die dem folgenden Beispiel zugrundeliegende Tabelle wurde mit **Code 13** erstellt:

CODE 13: SQL-CODE ZUR ERZEUGUNG VON TAB. 10

```
CREATE TABLE Welt(
Parent VARCHAR2(30) REFERENCES Welt,
Name  VARCHAR2(30) PRIMARY KEY);

INSERT INTO Welt VALUES (NULL, 'Welt') ;
INSERT INTO Welt VALUES ('Welt', 'Europa') ;
INSERT INTO Welt VALUES ('Europa', 'England') ;
INSERT INTO Welt VALUES ('Europa', 'Niederlande');
INSERT INTO Welt VALUES ('Europa', 'Deutschland') ;
INSERT INTO Welt VALUES ('Welt', 'Asien');
INSERT INTO Welt VALUES ('Asien', 'Japan');
INSERT INTO Welt VALUES ('Asien', 'China');
INSERT INTO Welt VALUES ('Welt', 'Amerika');
INSERT INTO Welt VALUES ('Amerika', 'United States');
INSERT INTO Welt VALUES ('Amerika', 'Mexico');
INSERT INTO Welt VALUES ('Welt', 'Afrika');
INSERT INTO Welt VALUES ('Afrika', 'Ägypten');
INSERT INTO Welt VALUES ('Afrika', 'Morokko');
```

Parent	Name
	Welt
Welt	Europa
Europa	England
Europa	Niederlande
Europa	Deutschland
Welt	Asien
Asien	Japan
Asien	China
Welt	Amerika
Amerika	USA
Amerika	Mexiko
Welt	Afrika
Afrika	Ägypten
Afrika	Marokko

Tab. 10: Beispiel zur hierarchi schen Gliederung

Hier ein Beispiel ohne `ORDER BY` (links) und mit `ORDER BY`: (rechts):

```
SELECT Parent, Name
FROM Welt
CONNECT BY PRIOR Name = Parent
START WITH Parent IS NULL;
```

```
Welt
        Europa
                England
                Deutschland
                Niederlande
        Asia
                Japan
                China
        America
                United States
                Mexico
        Africa
                Egypt
                Morocco
```

```
SELECT Parent, Name
FROM Welt
CONNECT BY PRIOR Name = Parent
START WITH Parent IS NULL
ORDER BY Name;
```

```
        Afrika
        America
        Asia
                China
        Egypt
                England
        Europe
                Germany
                Japan
                Marocco
                Netherlands
                United States
World
```

Die `CONNECT BY PRIOR`-Klausel ersetzt die `Join`-Bedingung. Im Gegensatz zu dieser ist die Anordnung der Spaltennamen, die dem Schlüsselwort `PRIOR` folgen, von Bedeutung, da sie bestimmt, ob die Abfrage, beginnend an dem mit `START WITH` bezeichneten Knoten, aufwärts oder abwärts gerichtet ist.

Aufgrund der datenbankinternen Optimierung rekursiver Joins liefern diese sehr schnell Ergebnisse. So offensichtlich einfach die Anwendung unter ORACLE und wahrscheinlich auch den genannten anderen DBMS ist, können sie in der Mehrzahl verfügbarer DBMS nicht genutzt werden. Daher wurde versucht, dieses Konzept zumindest auf die GIS-Ebene zu transferieren. Dieser Prozess soll im folgenden Kapitel beschrieben werden.

4.2.2.6 Parallele Arrays

Das Verfahren ist an die soeben beschriebenen rekursiven Joins angelehnt und sowohl als Stored Procedure als auch innerhalb eines GIS durch Iteration darstellbar, nicht aber mit einfachem SQL. Diese Methode ist jedoch die einzige der unter **Kap. 4.2.2** beschriebenen, die innerhalb der Arbeit auf der GIS-Seite mit AVENUE und MAPOBJECTS getestet und implementiert werden konnte und sich in diesem Bereich als effektiv erwiesen hat.

Die algorithmische Umsetzung der abwärtsgerichteten Query unterscheidet sich dabei naturgemäß von der nach oben gerichteten. Abwärts lässt sich der Vorgang als Schneeball-Effekt beschreiben. Zunächst wird die FGWID des (angeklickten oder automatisch gefundenen – vgl. **Kap. 4.1.3.10**) Start-Abschnittes – z. B. des Gebietsauslasses – in eine „Eltern"-Liste eingefügt und dann für alle Elemente dieser Liste alle Oberlieger gefunden. Hat ein Element der „Eltern"-Liste keinen weiteren Oberlieger, so ist dieser Abschnitt eine Quelle. Der Vorgang wiederholt sich solange, bis alle Quellen gefunden sind. Hier der dazugehörende Code-Ausschnitt:

CODE 11: AVENUE-CODE ZUR SELEKTION ALLER OBERLIEGER MITTELS INDIZIERTER ARRAYS

```
Starts = {StartFGWID}
Querystring = "([FGWID]="+""+StartFGWID.asString+""+") or " 'GIS-Abfrage initiieren
while (true) 'erzwungener Durchlauf
  newStarts = {}
  for each s in Starts
    ind = 0
    while(ind <> -1)
      ind = FGWULIDs.findByValue(s) 'Position in der Unterliegerliste ermitteln
```

```
      if(ind = -1)then break end 'Quelle gefunden
      FGWID = FGWIDs.get(ind) 'nächsten Eltern-Abschnitt einlesen
      Querystring = Querystring + "([FGWID] = "+""+ FGWID+""+") or "
      newStarts.add(FGWID) 'GIS-Abfrage aufbauen
      for each l in {FGWIDs, FGWULIDs, Points}
        l.remove(ind) 'Listen aus Performance-Gründen „aufräumen“
      end
    end
  end
  Starts = newStarts '„Eltern“-Liste aktualisieren
  if(Starts.count = 0)then break end 'alle Oberlieger gefunden und Abbruch der Schleife
end
'Sql-basierte GIS-Abfrage vervollständigen
Querystring = Querystring.left(Querystring.count - (") or ").count))
Querystring = Querystring + "))"
'GIS-Abfrage stellen
theFtab.query(Querystring, theFtab.getSelection, #VTAB_SELTYPE_NEW)
```

Die aufwärtsgerichtete Query (Abfrage) wird entsprechend umgekehrt formuliert. **Code 12** stellt eine Modifikation von **Code 11** zur Selektion zwischen zwei Abschnitten dar.

4.2.2.7 Zusammenfassung und Einschätzung

Das mentale Modell dieses Ansatzes ist sehr intuitiv, jedoch in der Realisierung aus Sicht der Arbeit mit wesentlichen Mängeln behaftet:

- Zunächst muss festgestellt werden, dass es, aus inhaltlichen und Kostengründen nicht einfach ist, diese Methoden einem breiten Kreis von Anwendern zugänglich zu machen, wobei die zuletzt beschriebene zumindest im GIS gut einsetzbar ist. Darüber hinaus gerät die Ausführungsgeschwindigkeit, abgesehen von den nur in bestimmten Datenbanken wie z. B. ORACLE unterstützten rekursiven Joins, mit zunehmender Anzahl Datensätze in einen unakzeptablen Bereich.
- Ein anderes Problem des Modells angrenzender Listen besteht darin, dass die Spalten FGWID und FGWULID im Grunde redundant sind [JENNINGS 2001]. Versucht man also einen Wert in einer Spalte zu ändern, muss er auch in der anderen angepasst werden.
- Unterordnungen werden nicht abgebildet. Die „Autorität“ fließt in der Hierarchie abwärts. Mit dem Entfernen eines Gewässerabschnittes werden jedoch alle darunter (vom Gewässer aus darüber) liegenden Abschnitte vom Rest des Systems abgeschnitten. Es gibt natürlich Situationen, in denen dies erwünscht ist, wie z. B. bei Wasserscheiden. Das ließe sich jedoch durch andere Mechanismen einfacher abbilden (vgl. **Kap. 4.3.1**).

All diesen Ansätzen ist vor allem der wiederholt herausgestellte Hauptkritikpunkt gemeinsam, dass sie nicht mit dem SQL-Standard anwendbar sind. Daher ist es auch nicht möglich, durch Unter-Oberlieger-Beziehungen simple hierarchische Abfragen in einem GIS oder einer Datenbank zu organisieren.

4.2.3 Implementierung klassischer Flussordnungsprinzipien

4.2.3.1 Einführung

Die hierarchische Struktur geomorphologischer Systeme wie ein Abflussnetzwerk ist traditionell eine Herausforderung für verschiedene Forschungsrichtungen. Das Verständnis eines Einzugsgebietes als Wasser-, Energie- und Stoffverteilungssystem kann durch eine interne numerische Organisation stark unterstützt werden. Nach einem bestimmten Ordnungssystem (Flussordnungskonzept) können die verschiedenen Fließstrecken in Flussabschnitte zusammengefasst werden. Diesen Abschnitten werden sogenannte Flussordnungen in Form von dimensionslosen Parametern zugewiesen (**Abb. 5**). Untersuchungen zu Fließgewässernetzen gibt es seit vielen Jahren. So führten HORTON [HORTON 1945], und STRAHLER [STRAHLER 1964] eine Methode zur Ordnung solcher Netzwerke ein, die als Standard betrachtet werden kann. Später entwickelte SHREVE [SHREVE 1967] eine alternative Methode, die auf gewichteten Strommaßen basiert.

Aufgrund der Bedeutung dieser Ordnungskonzepte für fließgewässermorphometrische und wasserhaushaltsbezogene Fragestellungen wurden Methoden ihrer Extraktion aus dem geometrischen Datenbestand sowie zur Implementierung in das Managementsystem (vgl. **Kap. 4.3.1**) in die Arbeit aufgenommen.

4.2.3.2 Das Ordnungsprinzip von STRAHLER

Das Flussordnungssystem nach STRAHLER geht von den Quellflüssen mit der Ordnung 1 aus. Vereinigen sich zwei Flüsse gleicher Ordnung, so erhält der neue Abschnitt eine um eins höhere Ordnung. Fließt dagegen in einen Abschnitt höherer Ordnung ein Gewässer mit niedrigerer Ordnungszahl, dann bleibt die höhere Zahl bestehen. Im Allgemeinen ist die Anzahl von Abschnitten erster und zweiter Ordnung signifikant größer, als diejenige höherer Ordnungen (**Abb. 5**, **Tab. 11**).

Für Einzugsgebiete mit geringen geologischen Unregelmäßigkeiten können für die Gewässernetze empirische Beziehungen abgeleitet werden:

- Wird die Anzahl der Flussabschnitte der Ordnung u nach STRAHLER mit Nu bezeichnet, dann bilden die Zahlen N_1, N_2, ...N_n im Durchschnitt eine geometrische Reihe: $Nu+1 = \alpha Nu$. Eine geometrische Reihe ist folgendermaßen definiert [BAUMGARTNER & LIEBSCHER 1990]:
  - Der Quotient aufeinanderfolgender Glieder ist konstant: $a_{n+1} / a_n = q$
  - Bildungsgesetz: $a_n = a_1 * q_{n-1}$
  - Jedes Glied ist das geometrische Mittel seiner Nachbarglieder: $a_n = \sqrt{a_{n+1} \cdot a_{n-1}}$

Die Zahl $R_b = N_i/N_{i-1}$ wird als Verzweigungsverhältnis (Bifurkationsindex, Maß der Multiplizität) bezeichnet [HAHN ET AL. 2000]. Ein Beispiel zeigt **Tab. 11**:

ORDNUNG	**N_i**	**R_b**
1	23	-
2	10	2.3
3	3	3.33

Tab. 11: Beispiel zur Berechnung des Bifurkationsindex nach STRAHLER

Ein Algorithmus zur Erfassung der STRAHLER-Ordnung muss entsprechend der obigen Beschreibung der Regel folgen, dass Quellen die Ordnung 1 erhalten und alle anderen

Abschnitte die kleinste der einmündenden Ordnungen bekommen. Dabei sind drei Wege sichtbar:

- von der Mündung zu den Quellen,
- von einem beliebigen Punkt innerhalb des Gewässernetzes aus und
- von den Quellen zur Mündung.

Innerhalb der Arbeit wurden der zweite und der dritte Ansatz untersucht. Der in **Code 14** vorgestellte Programmausschnitt basiert auf einem Algorithmus von K. LANFEAR [LANFEAR 1990]. Er hat den Vorteil, dass er auch auf Kreisstrukturen (z. B. Braided-River) angewandt werden kann, wenngleich dies die vorherige korrekte Ausrichtung der Abschnitte und ihre Belegung mit From- und ToNodes voraussetzt (vgl. **Kap. 4.1.1**). Er wurde hier auf die Verwendung von FGWIDs und FGWULIDs hin modifiziert. Allerdings steigt die Prozesszeit mit zunehmender Anzahl von Abschnitten – bei mehreren tausend Gewässersegmenten in einen unakzeptablen Bereich.

Im folgenden wird deshalb eine eigene Lösungsvariante von den Quellen zur Mündung in ihren Grundzügen beschrieben. Der gedankliche Ansatz besteht darin, das Gewässernetz von den Quellen her systematisch zu erfassen. Alle Quellabschnitte bilden die erste Ordnung.

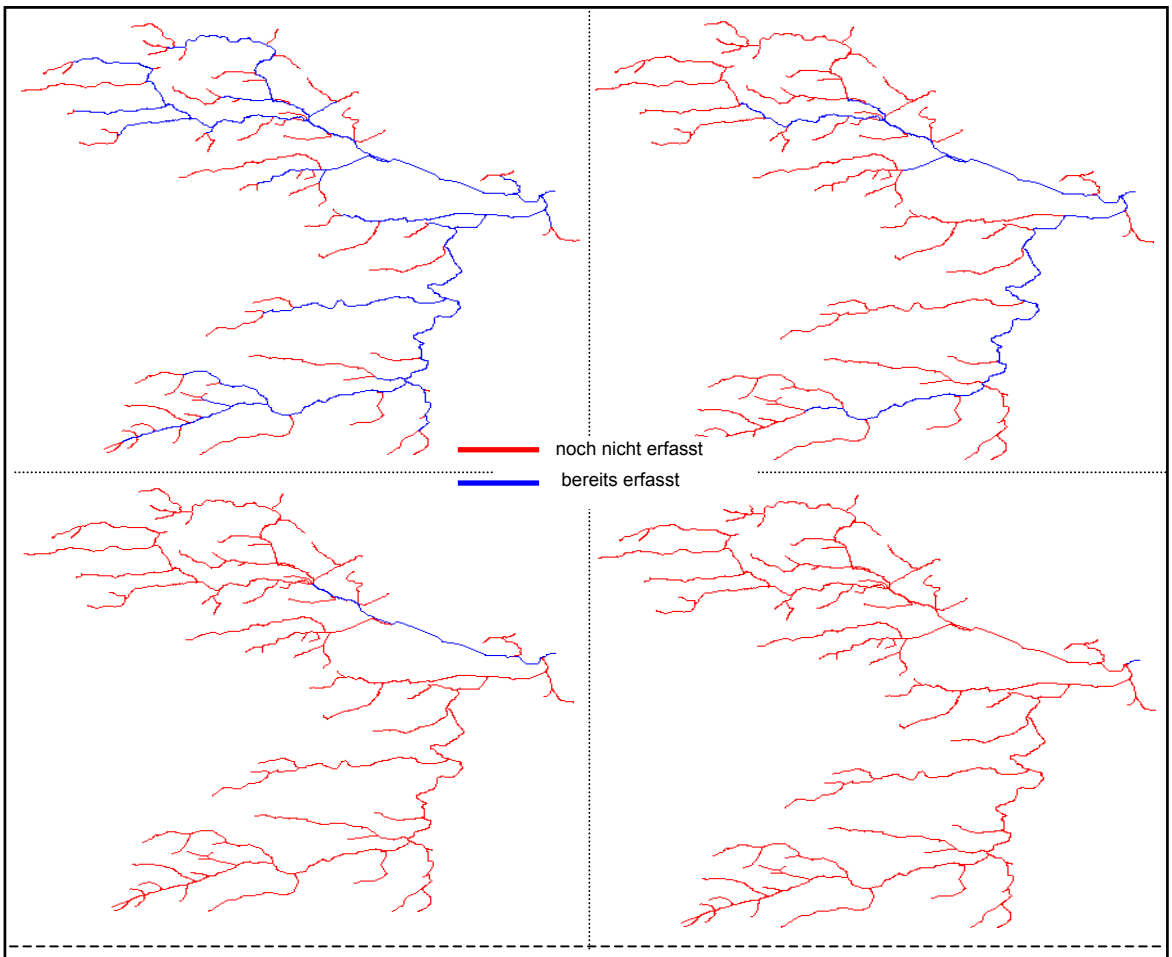


Abb. 47: Visualisierung des Algorithmus zur Erfassung der Strahler-Ordnung. Von links oben nach rechts unten sind die Ergebnisse nach jeweils einem weiteren Schleifendurchlauf dargestellt (SALZA)

Analog **Abb. 47** entspricht hierbei jeder Durchlauf einer Erhöhung der Ordnungsnummer. Dabei werden zunächst alle Abschnitte zwischen zwei Bifurkationen zu einem Strang in einer Liste zusammengefasst. Diese Liste beinhaltet pro Abschnitt eine weitere Liste mit

den FGWIDs sowie derjenigen FGWULID, die innerhalb dieses Stranges durch keine FGWID eines anderen hierzu gehörenden Elementes repräsentiert wird. Für diese ist klar, dass sie in einem anderen Strang gesucht werden muss. Sind alle Elemente in Strängen verteilt (eindeutige Beziehung), können sukzessive die Ordnungen von Außen nach Innen aus dem Gesamtbestand entfernt und damit bestimmt werden. Dass also ein Strang mit seinen Abschnitten entfernt wird, hängt davon ab, ob für ihn noch ein Oberliegerstrang existiert.

Es hat sich gezeigt, dass dieser Algorithmus aufgrund des Handlings in Clustern von Abschnitten sehr schnell und zuverlässig arbeitet – auch für große Gewässernetze.

Innerhalb der Datenbank könnte eine einfache Abfrage auf alle Abschnitte z. B. der dritten Ordnung, deren Durchfluss größer als 100 ist, wie folgt formuliert werden:

```
SELECT * FROM FGW WHERE Durchfluss > 100 AND Strahler = 3;
```

auf alle Abschnitte einer Strecke[28]:

```
SELECT * FROM FGW WHERE STRECKE = 33;
```

auf die Flussdichte:

```
d_Fl = SELECT SUM (1/A_EO* LENGTH);
```

auf die Flusshäufigkeit:

```
n = SELECT COUNT(STRECKE) FROM FGW;
h_Fl = n / A_EO
```

auf das Verzweigungsverhältnis:

```
n  = SELECT COUNT(*) FROM FGW WHERE ORNDNUNG=1;
n_1 = SELECT COUNT(*) FROM FGW WHERE ORNDNUNG=2;
R_b = n / n_1
```

Die Abbildung der Strahlerordnung in der Datenbank bietet also eine sehr schnelle Möglichkeit, unter anderem auch Informationen zur Geomorphologie über das betrachtete Gewässer zu beziehen, z. B. zur Ermittlung des Bifurkationsindex oder der Flussnetzweitefunktion, die wichtige Charakteristiken für Hochwasservorhersagen darstellen [DYCK & PESCHKE 1995]. Sie erfordert primär ein Feld STRAHLER und ggf. Attribute wie LÄNGE und STRECKE.

4.2.3.3 Die KLASSISCHE FLUSSORDNUNG

In der KLASSISCHEN FLUSSORDNUNG wird die 1 dem Hauptfluss zugeschrieben. Alle in diesen mündenden Flüsse erhalten die zweite Ordnung usw. (**Abb. 5**). Die Gewässer Deutschlands wurden beispielsweise nach diesem System organisiert, wobei die Funktion eines Flusses und einige andere Aspekte von Bedeutung, wie Wasserführung, Länge, Größe des Einzugsgebietes, Höhenlage der Quelle usw., in diese Zuweisung einbezogen werden [LEHNER 1998].

Das klassische Flussordnungsprinzip wurde 1998 von B. LEHNER (LEHNER 1998) als Grundlage eines dreistufigen Modells aus einem Vernetzungs-, einem Ordnungs- und einem Kodierungskonzept betrachtet und im GIS SPANS© implementiert.

[28] von einer Quelle bis zum Gebietsauslass

Dort sollten aus der zweidimensionalen Geometrie des Gewässernetzes die hierarchischen Zusammenhänge in Form einer Kombination der Ordnung mit Entfernungsangaben in einer Tabelle gespeichert werden und durch SQL-Statements zu nutzen sein. Grundlage dieses Ansatzes ist die KLASSISCHE FLUSSORDNUNG (**Abb 48**). Es zeigt sich jedoch, dass das System relativ speicherintensiv und redundant ist, da mit der Anzahl der im Gewässernetz gefundenen Flussordnungen die Zahl der benötigten Felder „n.ORD“ steigt (vgl. **Kap. 2.1.5**). Wichtigster Kritikpunkt ist wohl der Umfang der den SQL-Statements zugrunde liegenden Stringoperationen, da diese erst aus relativ vielen Zeilen und Spalten zusammengebaut werden müssten. Interessant ist jedoch, dass sich dieses Prinzip auch auf andere Ordnungen, wie die nach STRAHLER oder SHREVE aufbauen lässt. Die KLASSISCHE FLUSSORDNUNG kann einen guten Überblick zur längenabhängigen Struktur des Gewässernetzes geben. Der Algorithmus wurde hier nach dem folgenden Ablaufschema entworfen:

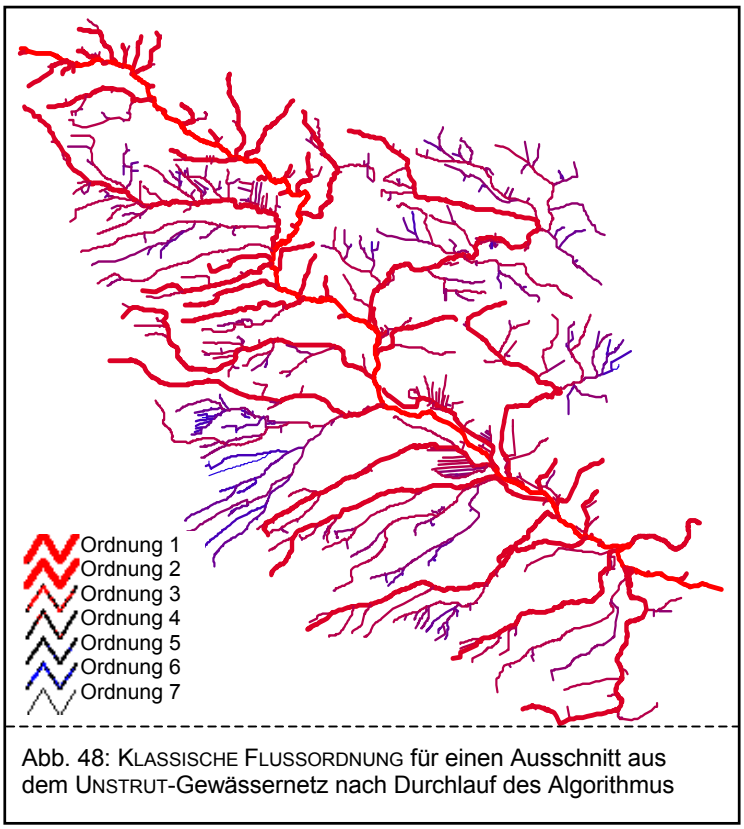


Abb. 48: KLASSISCHE FLUSSORDNUNG für einen Ausschnitt aus dem UNSTRUT-Gewässernetz nach Durchlauf des Algorithmus

Zunächst werden auf Grundlage von **Code 6** die Gewässerabschnitte ausgerichtet und ihre Länge bis zu den Quellen aufsummiert (kumulatives Attribut, dieses können jedoch auch der Durchfluss oder z. B. die Größe des Einzugsgebietes sein). Im zweiten Schritt wird pro Cluster (vgl. **Kap. 4.1.3.7**) jeweils die Quelle mit dem längsten Abschnitt selektiert und von der Quelle bis zur Mündung des Clusters alle auf dieser Strecke liegenden Abschnitte als Ordnung 1 (Attribut KLA_ORD) beschrieben. Von den Bifurkationen ausgehend, die die Abschnitte erster Ordnung berühren, werden die Segmente auf Basis von Code 11 zu oberliegenden Clustern geordnet (in ein Array gelesen). Aus jedem die-ser Cluster wird nun die Quelle mit dem höchsten Wert im kumulativen Attribut als Ausgangspunkt für die Bewegung in Richtung Gebietsauslass gewählt. Diese Aufwärtsbewegung erfolgt auf dem kürzesten Weg entlang der ausgerichteten Abschnitte solange, bis ein Abschnitt erster Ordnung gefunden ist. Alle bis dahin auf diesem Weg liegenden Abschnitte erhalten die Ordnung 2. Der Vorgang wird solange wiederholt, bis alle Abschnitte mit einer KLASSISCHEN FLUSSORDNUNG belegt sind (**Abb. 48**).

4.2.3.4 HORTONS Flussordnung

R.E. HORTON [HORTON 1945] wandte morphometrische Analysen auf eine Auswahl von Gewässerattributen an und führte diese Untersuchungen in verschiedenen Gesetzen zur

Abflussgestalt zusammen. HORTONs Gesetz der Flusslängen nimmt an, dass eine geometrische Beziehung zwischen der Anzahl der Flusssegmente in wachsenden Ordnungen besteht und die Flächen der Einzugsgebiete von sukzessive geordneten Gewässerabschnitten eine lineare Beziehung bilden („Law of Basin Areas“). Studien anderer natürlicher Baumstrukturen haben ähnliche Muster gezeigt. Beispielsweise wurde im Stoffverteilungssystem von Bäumen, den Verästelungsstrukturen von Blättern oder dem menschlichen Blutkreislauf ein Bifurkationsverhältniss von Drei entdeckt [JING ET AL. 1998], [DODDS & ROTHMAN 2000]. Die Implementierung erfolgt durch eine Modifikation des Algorithmus zur Erfassung der Klassischen Flussordnung.

4.2.3.5 Das System von SHREVE

Das System von SHREVE schätzt die Durchflussrate (Stream Discharge) und das gewichtete Strommaß (Stream Power) eines Segmentes im Gewässernetz. Die SHREVE Link Magnitude weist jedem Abschnitt die Anzahl von Zuflüssen erster Ordnung (Quellen) oberhalb zu und korreliert auf diese Art Stream Discharge mit Stream Power [CHENG ET AL. 2001] (**Abb. 49**). Sie kann als (Schätz-) Maß für das Wasservolumen oder andere quantitative Werte herangezogen werden, sagt jedoch nichts über den relativen Beitrag eines Flusssegmentes zum Gesamt-Abflussvolumen (ein Fluss erster und einer der zweiten Ordnung könnten die gleiche Durchflussrate aufweisen). Diese Ordnungsnummer ist auch nicht notwendigerweise proportional zu Einzugsgebietsgröße oder Flussquerschnitt.

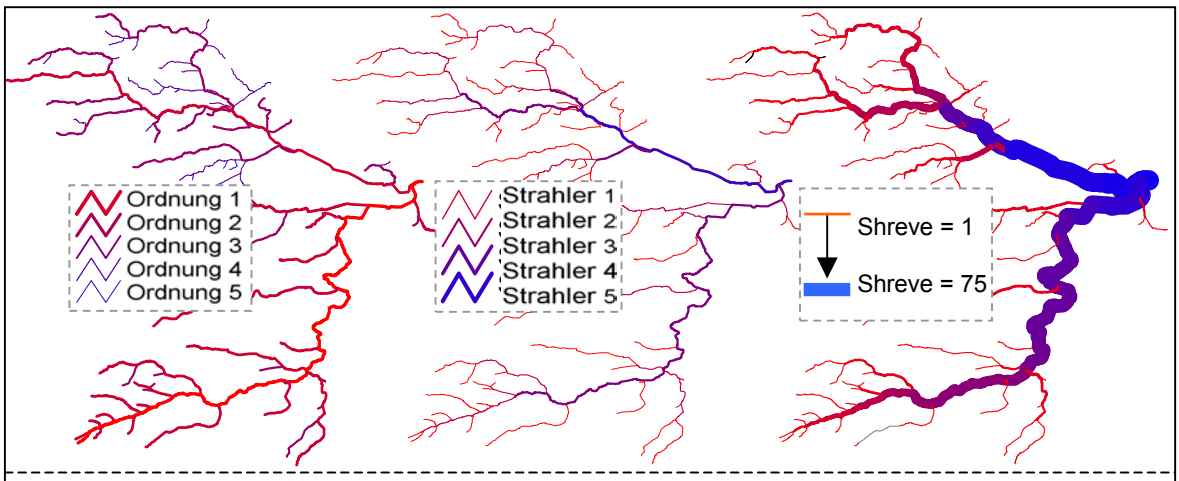


Abb. 49: Von links nach rechts: KLASSISCHE FLUSSORDNUNG, STRAHLER- und SHREVE-Ordnung für die SALZA

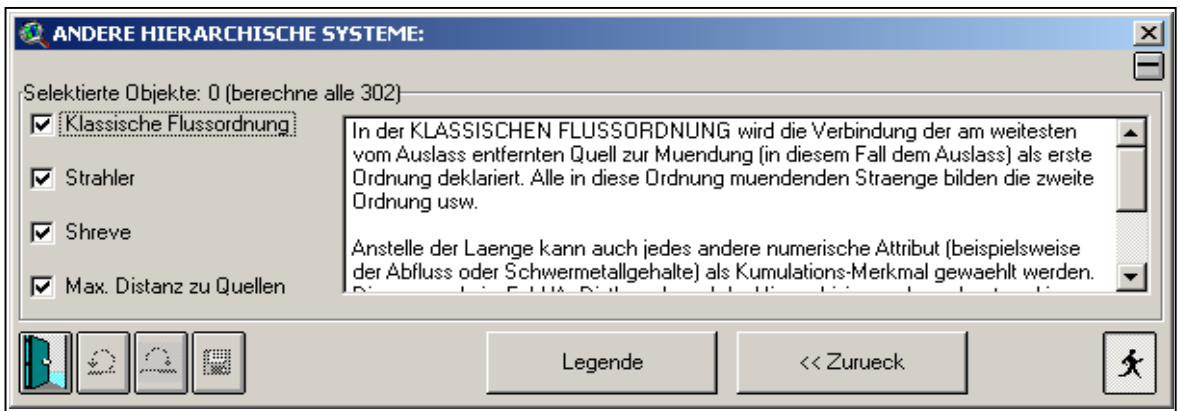


Abb. 50: Benutzerinterface zur Erfassung klassischer Flussordnungskonzepte innerhalb des GIS

Die Extraktion dieser Ordnung erfordert also die Anzahl aller oberhalb eines Abschnittes gelegenen Quellen. In folgenden Abschnitten werden Codierungssysteme vorgestellt, die unter anderem auch dafür eine schnelle und SQL-basierte Lösung anbietet. Die GIS-Lösung stützt sich wieder auf **Code 11**, ermittelt aus dem damit gefunden Cluster die Anzahl von Quell-Abschnitten und weist diese dem betrachteten Gewässersegment zu.

4.2.3.6 Zusammenfassung und Einschätzung

Die hier beschriebenen klassischen Codierungssysteme stammen alle aus dem „prä-PC“-Zeitalter und sind primär morphometrisch angelegt. Sie beschäftigen sich mit der mathematischen und statistischen Auswertung dieser Ordnungen, um daraus dem Gewässernetz innewohnende Eigenschaften abzuleiten (z. B. in der experimentellen Geomorphologie, vgl. [RINALDO & RODRIGUEZ-ITURBE 2001]). Sie können dementsprechend auch in einem datenbankgekoppelten Managementsystem zur Abschätzung morphometrischer Parameter eingesetzt werden (vgl. **Code 14**). Aus diesem Grund wurde ihre automatisierte Erfassung angestrebt und in die Gesamtapplikation implementiert (**Abb. 50**). Für flussab- oder aufwärtsgerichtete SQL-Statements sind sie jedoch nicht bzw. nur eingeschränkt geeignet.

4.2.4 Der WEG-Schlüssel der Länderarbeitsgemeinschaft Wasser (LAWA)

4.2.4.1 Einführung

Die Länderarbeitsgemeinschaft Wasser wurde 1956 als Zusammenschluss der für die Wasserwirtschaft und das Wasserrecht zuständigen Ministerien der Länder der Bundesrepublik Deutschland gebildet. Ziel war es, auftauchende wasserwirtschaftliche und wasserrechtliche Fragestellungen gemeinsam zu erörtern, Lösungen zu erarbeiten und Empfehlungen zur Umsetzung zu initiieren. Sie hat im Dezember 1970 die "Richtlinie für Gebietsbezeichnungen" beschlossen. Dabei erstrecken sich diese Gebiete meist über mehrere Bundesländer. Deshalb wird bei der Verschlüsselung eine bundeseinheitliche Systematik verwendet, die eine Bezeichnung der Gebiete oder darin abgebildeter Teileinzugsgebiete und Objekte in Form von Gebietskennzahlen und deren Aktualisierung bzw. Fortschreibung ermöglicht und für eine automatisierte Datenverarbeitung geeignet ist.

Aus hydrologischer Sicht ist das Gebiet der Bundesrepublik Deutschland in sechs Stromgebiete (DONAU, RHEIN, EMS, WESER, ELBE und ODER) und die Küstengebiete aufgeteilt und wird durch Zahlen gekennzeichnet. Dabei liegt die Unterteilung der Strom- und Küstengebiete in Teileinzugsgebiete sowie die Darstellung in Kartenwerken und Verzeichnissen entsprechend den Gebietsanteilen der Einzugsgebiete in der Verantwortung der jeweiligen Bundesländer.

Die Gebietskennzahl besteht aus maximal 10 Ziffern. Buchstaben oder Sonderzeichen werden nicht verwendet. Um die Kodierung konsistent zu halten, kann die 0 zum Auffüllen der vollen 10-stelligen Zahlen verwendet werden. Die erste Stelle der Gebietskennzahl bezeichnet das Stromgebiet, zu dem das Einzugsgebiet gehört: 1=DONAU, 2=RHEIN, 3=EMS, 4=WESER, 5=ELBE, 6=ODER. Die Küstengebiete von Nord- und Ostsee erhalten die 9 als erste Ziffer der Gebietskennzahl.

Durch die Unterteilung der Stromgebiete werden Teileinzugsgebiete gebildet. Auf Grundlage der Geländemorphologie bzw. der Vorfluterverhältnisse werden die Wasserscheiden

der oberirdischen Einzugsgebiete festgelegt. Flächen ohne oberirdischen Abfluss werden denjenigen Fließgewässern zugeordnet, denen auf Grund der Wasserscheiden theoretisch der Niederschlag zufließen würde. Die zweite Stelle der Gebietskennzahl gibt die erste Untereilung des oberirdischen Einzugsgebietes des jeweiligen Stromgebietes an. Die folgenden Stellen unterteilen die Teileinzugsgebiete weiter. Diese werden durch die Wasserscheiden begrenzt, die von den einmündenden, durch gerade Ziffern gekennzeichneten Fließgewässer ausgehen. Unter dem Gesichtspunkt der Zweckmäßigkeit kann in Ausnahmefällen die unterteilende Wasserscheide von einem anderen Punkt als der Einmündung in das Hauptfließgewässer ausgehen. Dabei sollten dann markante Punkte wie z. B. Brücken, Sperrdämme oder Pegel bevorzugt werden [LAWA 1978].

Mit dem Quellgebiet bei Ziffer 1 beginnend, werden bis zur Mündung grundsätzlich neun Teileinzugsgebiete festgelegt. Die Einzugsgebiete entlang des Hauptfließgewässers werden als Zwischengebiete bezeichnet und erhalten die ungeraden Ziffern 3, 5, 7 und 9. Einzugsgebieten der Nebenfließgewässer werden gerade Ziffern zwischen 1 und 10 zugeschrieben. Die Aufteilungen von Zwischengebieten erfolgen analog dieser Systematik. Die Festlegung der Nebenfließgewässer berücksichtigt auch ihre Einzugsgebietsgröße. Dabei sollten die vier flächenmäßig größten Nebenfließgewässer bei jeder Stelle der Gebietskennzahl herangezogen werden. In sehr unterschiedlich großen Zwischengebieten können auch kleinere Nebenfließgewässer zur Unterteilung einbezogen werden. Die Gebietskennzahl des untersten Zwischengebietes endet stets auf der Ziffer 9. Insbesondere bei kleinen Teileinzugsgebieten kann es sinnvoll sein, in weniger als neun Teileinzugsgebiete zu unterteilen. In diesen Fällen können ab der vierten bis zur zehnten Stelle die Ziffern 8, 7 oder noch weitere ausgelassen werden.

Die Gewässerkennzahl ist für das gesamte Einzugsgebiet des Fließgewässers - von der Quelle bis zur Mündung - mit der Gebietskennzahl identisch. Demnach kann sie aus einer Kennzahl aus bis zu zehn Ziffern bestehen. Hier ein Beispiel:

- ELBE 5
    - SAALE 56
        - SALZA 567251
            - QUERNE (Quelle) 5672111

Da diese Kodierung wissensbasiert und primär an den Einzugsgebieten orientiert ist, kann der WEG-Schlüssel nicht automatisch zugewiesen werden. Abfragen zur Gewässerhierarchie können jedoch auf der Analyse der Anlage dieser Kodierung basieren. Auch hier gibt es wieder zwei Hauptformen der hierarchischen Abfrage:

- abwärtsgerichtet: „Selektiere alle Abschnitte, die in diesen Punkt münden..."
- aufwärtsgerichtet: „Selektiere alle Abschnitte auf dem kürzesten Weg bis zum Gewässerauslauf..." und ihre Sonderform:
    - „Selektiere alle Abschnitte zwischen zwei Punkten auf dem kürzesten Weg..."

Diese Abfragen erfordern unterschiedliche, nachfolgend beschriebene String- (Zeichenketten) Behandlungen.

4.2.4.2 Die abwärtsgerichtete Abfrage

Sollen alle Gewässerabschnitte oberhalb eines bestimmten Punktes ausgewählt werden, so sind zunächst die Fragen zu stellen:

- was allen diesen Abschnitten gemeinsam ist und
- was sie darüber hinaus von allen übrigen Abschnitten unterscheidet.

Exemplarisch sei die Vorgehensweise am Beispiel der oberen Querne dargestellt **(Abb. 51**).

Die WEG-Schlüssel der zu findenden Abschnitte sind:

```
5672110000                          Andere, nicht in den gesuchten
5672111000                          Filter passende Abschnitte in
5672113000                          der direkten Umgebung haben z. B.
5672112000                          folgende Schlüssel:
5672114000                          5672118000
5672115000                          5672131000
5672116000                          5672132000
```

Zwei Aspekte sind hier von Bedeutung:

- Die gesuchten WEG-Schlüssel sind kleiner oder gleich dem des Start-Abschnittes 5672117000.
- Die erste gerade Zahl, die angetroffen wird, wenn man den Start-Code von rechts nach links liest, nimmt eine besondere Rolle ein. Bei der 5672117000 – ist dies die Ziffer 2. Ersetzt man alle Ziffern, die rechts davon stehen durch eine Null, so ergibt sich die Zahl 5672000000. Alle gesuchten Abschnitte sind größer oder gleich dieser Zahl.

Der Algorithmus zur Untersuchung und Veränderung des Start-Codes im Sinne des zweiten Punktes sei an dieser Stelle kurz beschrieben:

CODE 15: JAVA-FUNKTION ZUR ERMITTLUNG DES KLEINSTEN STARTWERTES FÜR DEN WEG-SCHLÜSSEL BEI DER AUFWÄRTSGERICHTETEN ABFRAGE

```
Public String findeWEG(String WEG){  //Deklaration der Funktion mit Parameterübergabe (Start-Code)
int k=0; //Initalisierung der Variablen
String Rueckgabe="", xx = "";
for (int x=(WEG.length()-1); x > 0; x--){  //Zeichenkette von rechts nach links
          if(k!=0) continue; //Null ignorieren
          xx = WEG.substring(x, x-1 ); //Ziffer zurückgeben
          if(xx != "0"){  //Erste gerade Zahl finden und alle bis dorthin gefundenen Zahlen durch Null ersetzen
            if(Integer.valueOf(xx) mod 2 != 0) Rueckgabe = "0" + Rueckgabe;
            else Rueckgabe = xx + Rueckgabe;
            k = x;
          else  Rueckgabe = "0" + Rueckgabe;
        }
       Rueckgabe = WEG. substring (0, k)+ Rueckgabe; //Rückgabestring
       return Rueckgabe; //Verlassen des Algorithmus und Rückgabe an die aufrufende Funktion / Klasse
}
```

Jede Bedingung für sich genommen würde die Grundgesamtheit aller betrachteten Abschnitte nicht ausreichend einschränken. In der Kombination werden jedoch genau die gewünschten Schlüssel und somit an sie gebundene Objekte extrahiert. Als SQL-Anweisung würde die aus dieser Analyse abgeleitete SQL-Abfrage

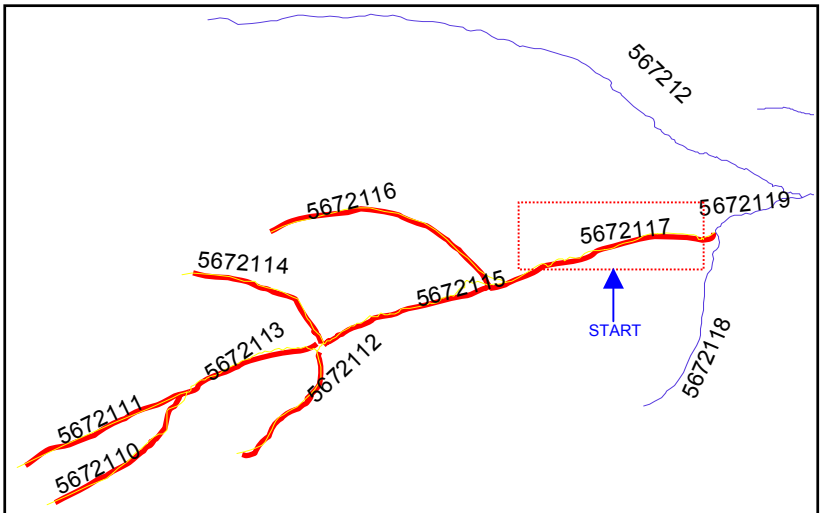


Abb. 51: WEG-Schlüssel der oberen QUERNE –abwärts gerichtete Query

allgemein formuliert wie folgt aussehen:

```
SELECT *
FROM Attributtabelle
WHERE FELD_WEG >= Rueckgabe and FELD_WEG <= WEG;
```

Ein spezielles Beispiel wäre:

```
SELECT *
FROM Attributtabelle
WHERE FELD_WEG >= 5672000000 and FELD_WEG <= 5672117000;
```

Der SQL-Abfrage müsste immer die soeben beschriebene String-Verarbeitung vorausgehen. Da diese jedoch nur eine Zeichenkette (die des Start-WEG) betrifft, ist der Zeitaufwand unerheblich.

4.2.4.3 Die aufwärtsgerichtete Abfrage

Auch hier stellen sich bei der Analyse wieder die beiden Fragen

- Was ist allen diesen Abschnitten gemeinsam?
- Was unterscheidet sie darüber hinaus von allen übrigen Abschnitten?

Die WEG-Schlüssel der zu findenden Abschnitte sind:

```
5672110000(Start)          Andere, nicht in den gesuchten
5672113000                 Filter passende Abschnitte in
5672115000                 der direkten Umgebung haben z. B.
5672117000                 folgende Schlüssel:
5672119000                 5672364000
5672133000                 5672160000
5672131000                 5672132000
5672139000                 5672132100
```

Als zunächst augenscheinlichste Gemeinsamkeit kristallisiert sich die Zahl `56721` heraus. Diese ist jedoch auch in einem Teil der unerwünschten Abschnitte enthalten. Der Kern der Lösung liegt auch hier wieder in der ersten geraden Zahl von rechts – im Falle der `5672110000` ist dies die 2. Bei näherer Betrachtung der erwünschten Codes wird klar, dass bei ihnen von rechts nach links bis zu dieser Zahl keine gerade Ziffer auftritt. Es müssten also folgende Fragen gestellt werden:

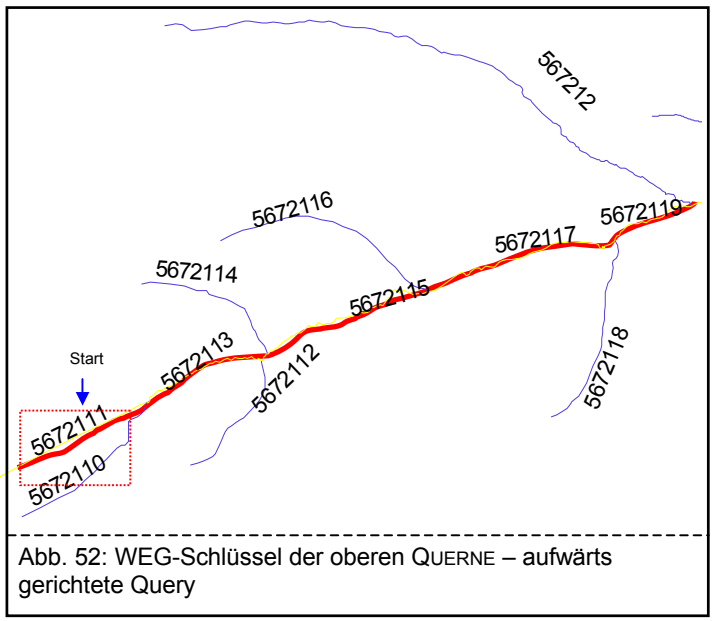


Abb. 52: WEG-Schlüssel der oberen QUERNE – aufwärts gerichtete Query

- An welcher Stelle – sukzessive von rechts betrachtet – tritt beim Start-WEG-Schlüssel die erste gerade Zahl auf? Im Beispiel ist dies die siebente Stelle.
- Welche WEG-Schlüssel haben – ebenfalls schrittweise von rechts betrachtet – bis zu dieser Stelle keine gerade Zahl?
- Welche WEG-Schlüssel sind bis zu dieser Stelle – nun jedoch von links betrachtet – gleich dem Start-WEG? (**5672**)?

- Welche Schlüssel sind größer, als der Start-Schlüssel?

Diese vier Fragen lassen sich leider in dieser Form nicht in einer einzigen SQL-Anweisung abbilden. Deshalb wird die Abfrage durch einen einzigen Schleifendurchlauf unterstützt. Die Umsetzung im Programm stellt sich wie folgt dar:

CODE 16 JAVA-FUNKTION ZUM AUFFINDEN DER UNTERLIEGER EINES BESTIMMTEN WEG-SCHLÜSSELS

```
Public String[] findWEGDown(String WEG){ //Deklaration der Funktion mit Parameterübergabe (Start-Code)
   public String selectDownStream(String Auslauf){//Variablen-Deklaration
   String WEG = theTable.returnValueString(wegField, theKey);
   String WEGs[] = String[theTable.getNumRecords()];
   String astr = "";
   Int a = 0;
   theTable.MoveFirst();//Record zurücksetzen
   while(Not(theTable.EOF)){ //Durchlauf der Tabelle
     String w = theTable.Value(wegField);
     if(WEGs.find(w)<>-1)continue;//Übergehen, wenn Schlüssel bereits in der Liste ist
     if((Double)w<(Double)WEG){continue};//Casting des Schlüssels und Vergleich mit WEG, ggf. übergehen
     Take = compareParts(WEG, w); //Übergabe an Funktion zur String-Analyse (CODE 3.2.9)
     if(Take) WEGs[a] = w; //zum Array der WEGs hinzufügen
     theTable.MoveNext();
   }
   return Sort(WEGs, True); //Rückgabe der aufsteigend sortierten WEG-Schlüssel
}
```

Code 16 findet dabei die erste gerade Ziffer – sukzessive von rechts aus betrachtet - und vergleicht dann alle links davon befindlichen Zeichen auf Gleichheit mit dem linken Teil des Start-Abschnittes.

Die Query für das oben beschriebene Beispiel (`Start-WEG = 5672110000`) würde dann wie folgt aussehen (vgl. auch **Abb. 52**):

```
SELECT *
FROM Tabelle
WHERE WEG IN
('5672111000',  '5672113000',  '5672115000',  '5672117000',  '5672119000',  672131000',
'5672133000',  '5672139000',  '5672150000',  '5672170000');
```

Die Komplexität der `WHERE`-Klausel ist an dieser Stelle unerheblich, da sie durch das Programm automatisch generiert wird[29].

4.2.4.4 Abfrage der direkten Verbindung entlang des Gewässers

Eine Abfrage der direkten Verbindung entlang des Gewässers erweitert die zuvor beschriebene Fragestellung um

- eine weitere aufwärtsgerichtete Abfrage und
- den Ausschluss unerwünschter Abschnitte - der Duplikate beider aufwärtsgerichteter Abfragen (**Abb. 52**, **Abb. 53**).

Es ist also eine doppelte aufwärtsgerichtete Abfrage mit einem zusätzlichen Filter. Der Charakter

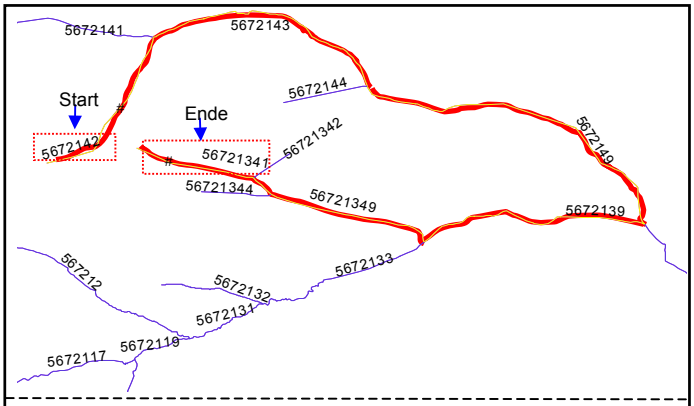


Abb. 53: WEG-Schlüssel – direkte Verbindung entlang des Gewässers (QUERNE-WEIDA)

[29] Allerdings hat es sich bei den Tests unter MAPOBJECTS gezeigt, dass dort eine so formulierte Query zu erheblichen Performace-Problemen führen kann, da offenbar jeder Eintrag in der Liste separat behandelt wird.

dieses Filters wird beim Betrachten der Aufgabenstellung in **Abb. 53** deutlich. Es geht darum, alle Abschnitte zu finden und vom Ergebnis zu separieren, die im Resultat beider Querys erscheinen würden – also das „kleinste gemeinsame Vielfache“. **Code 16** wurde in diesem Sinne modifiziert.

Die Abfrage für das oben beschriebene und abgebildete Beispiel (Start-WEG = `5672141000`, End-WEG = `5672160000`) würde dann wie folgt aussehen:

```
SELECT *
FROM Tabelle
WHERE [Weg] IN ('5672141000', '5672143000', '5672149000', '5672160000');
```

4.2.4.5 Zusammenfassung und Bewertung

Mit dem WEG-Schlüssel wurde eine bundeseinheitliche Systematik zur Bezeichnung von Einzugsgebieten oder darin gebildeter Teileinzugsgebiete eingeführt und mittlerweile in vielen Bereichen des Gewässermonitorings und der Gewässerbewirtschaftung eingesetzt. Die Gebietskennzeichnung erfolgt durch zehnstellige Gebietskennzahlen, die eine numerische Verschlüsselung der oberirdischen Einzugsgebiete der Fließgewässer Deutschlands darstellen. Sie sind allerdings wissensbasiert und können demnach nur vom Bearbeiter durch sein Fach- und Regionalwissen zugeordnet werden.

Die stromauf- und vor allem die stromabwärtsgerichteten Querys lassen sich nur mit relativ komplexen Stringoperationen formulieren. Da sie dem beschriebenen Standard entsprechen, in weiten Teilen bereits vorhanden und damit von nationaler Bedeutung sind, wurden sie dennoch in dieser Arbeit beschrieben und implementiert.

Der WEG-Schlüssel lässt sich sowohl im GIS als auch im RDBMS / RDB nutzen. Eine bereits im Einsatz befindliche Anwendung dieser Untersuchungen (vgl. **Kap. 4.3.1**) dient der automatisierten Qualitätskontrolle vergebener Codierungen mithilfe clusterbezogener, von allen Quellen ausgehender aufwärtsgerichteter Abfragen. Die dabei nicht erfassten Abschnitte weisen fehlerhafte Schlüssel auf.

Die bisher dargestellten Möglichkeiten zur Abbildung von Fließgewässerhierarchien sind im Sinne der Zielstellung der Arbeit nur eingeschränkt nutzbar (vgl. **Tab. 2**). Daher werden in den folgenden Kapiteln eigene, graphenbasierte Ansätze eines optimierten hierarchiebasierten Datenhandlings beschrieben.

4.2.5 Graphenbasierte Abbildung von Fließgewässerhierarchien

4.2.5.1 Einführung in die Graphentheorie

L. EULERs „Königsberger Brückenfrage“ von 1736 wird oft als der Beginn des mathematischen Routings betrachtet: ist es möglich, einen Weg durch Königsberg zu finden, der jede Brücke genau einmal überquert und dabei zum Ausganspunkt zurückkehrt [SPACCAMELA & NANNI 2000]? Dieses Gedankenmodell war der Beginn der Graphentheorie und bereitete den Weg für die Implementierung von Routing-Algorithmen in heutigen GIS-Applikationen, die sich mit räumlichen Netzen aller Art beschäftigen [SUKTHANKAR ET AL. 1992].

Die folgenden Ausführungen stützen sich auf Darstellungen in [ERWIG 1994], [ZIETLOW 1993], [RÖSCH 1998] und [FRIGIONI ET AL. 2000]. Unter einem Graph versteht man ein Netzwerk, in dem Knoten (Punkte, Nodes) durch Kanten (Geraden) verbunden werden (vgl. **Abb. 54**). Ein Graph G besteht demnach aus einer endlichen, nicht leeren Menge V von p Knoten, zusammen mit einer Menge E von p bidirektionalen Teilmengen von V. Man unterscheidet dabei benachbarte und nicht benachbarte Knoten. Bei $V=\{v_0, v_1, ..., v_m\}$, $E=\{e_1, e_2, ..., e_n\}$, ist $G=(V, E)$ ein Graph (vgl. **Abb. 54, Abb. 55** und **Abb. 56**). Die Kantenfolge eines Graphen ist dementsprechend die alternierende Folge von Ecken und Kanten $v_0, e_1, v_1, ..., v_n\text{-}1, e_n, v_n$, die bei $v_0 = v_n$ als geschlossen bezeichnet wird – hier läge also eine Ringstruktur vor (**Kap. 4.1.3.7**).

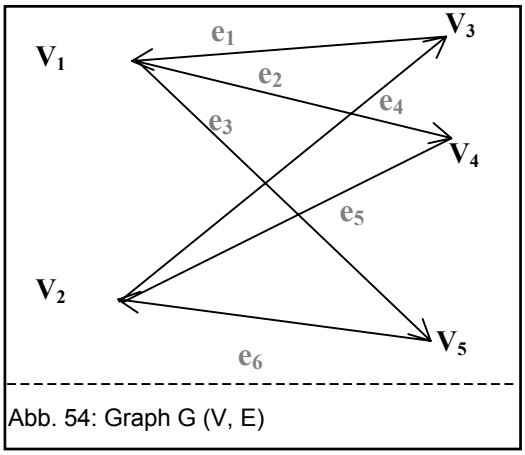


Abb. 54: Graph G (V, E)

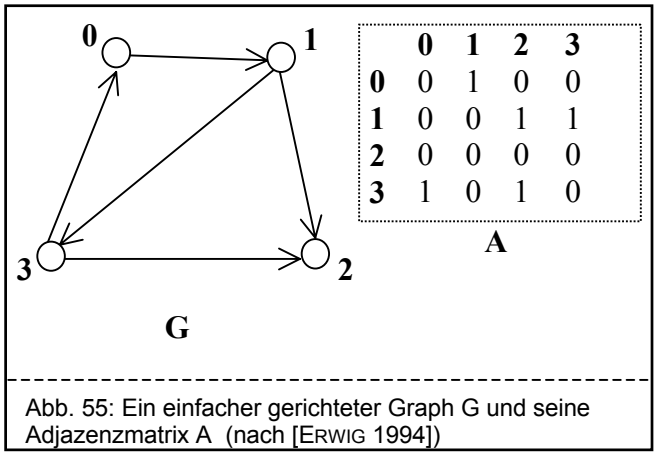


Abb. 55: Ein einfacher gerichteter Graph G und seine Adjazenzmatrix A (nach [ERWIG 1994])

Führt man neben den Begriffen Knoten und Kanten die Masche als Element ein, welche die von Kanten umschlossene Fläche bezeichnen soll, so ergibt sich daraus die bereits von EULER gefundene Beziehung für planare Graphen [MAHESHWARI & ZEH 2001]. Kanten von Graphen können bestimmte Eigenschaften haben, die den gesamten Graphen charakterisieren. So können Kanten beispielsweise gerichtet sein und dadurch nur in bestimmten Richtungen durchlaufen werden (gerichteter Graph) und gewichtet sein (beispielsweise nach der Länge oder dem Abfluss oder der Zeit...). Erreicht man einen beliebigen Knoten eines Graphen auf zwei unterschiedlichen Wegen, so enthält er Zyklen. Der Graph aus **Abb. 55** sowie Gewässernetze mit Ringstrukturen gehören dieser Gruppe an, die Graphen in **Abb. 56** hingegen nicht. Nachbarschaftsbeziehungen zwischen Elementen des Graphen (Knoten, Kanten und Maschen) werden mit den Begriffen Inzidenz und Adjazenz zum Ausdruck gebracht. Für den Graphen als Gesamtheit kann dies auch mithilfe der Matrizenschreibweise in Form der Inzidenzmatrix I bzw. Adjazenzmatrix A geschehen (vgl. **Abb. 55**) [DIESTEL & KÜHN 2003].

In Verbindung mit dieser Arbeit sind noch einige weitere Begriffe zu klären. Ein Graph heißt zusammenhängend, wenn es zu jeweils zwei verschiedenen Kanten mindestens einen Weg gibt. Diese Bedingung ist Voraussetzung für die Definition des Baumes, die besagt, dass ein zusammenhängender Graph, der keine geschlossenen Kantenzüge (Zyklen) enthält, als Baum bezeichnet wird (**Abb. 56,** in **Kap. 4.1.3.8** wurde der Begriff Cluster verwendet, vgl. dazu auch **Kap. 2.3**). Mehrere Bäume bilden einen Wald. Ein normalhierarchisches Gewässernetz kann demnach als Baum aufgefasst werden; mehrere nichtzusammenhängende Gewässernetzcluster stellen einen Wald dar.

Ein Baum besteht aus Knoten (Nodes, Bifurkationen), die über Kanten (Branches, Gewässerabschnitte) miteinander verbunden sind. Die Bezeichnungen der Strukturelemente eines Baums sind zum Teil der Natur oder der Nomenklatur von Familienstammbäumen entlehnt. Im folgenden sollen der allgemeine Aufbau eines Baumes und einige seiner wichtigsten Ausprägungen beschrieben werden (vgl. dazu auch [MEHLHORN 1986], [SEDGEWICK 1992] und [ZIETLOW 1993]).

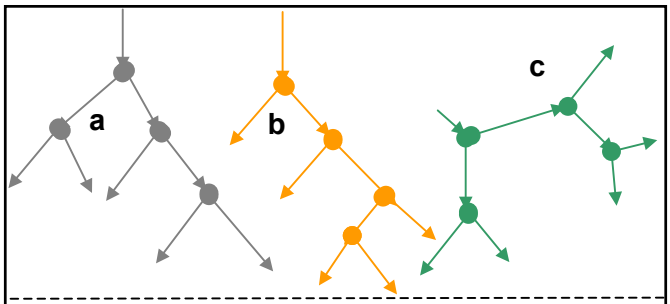


Abb. 56: Bäume im „Wald“ der Graphentheorie

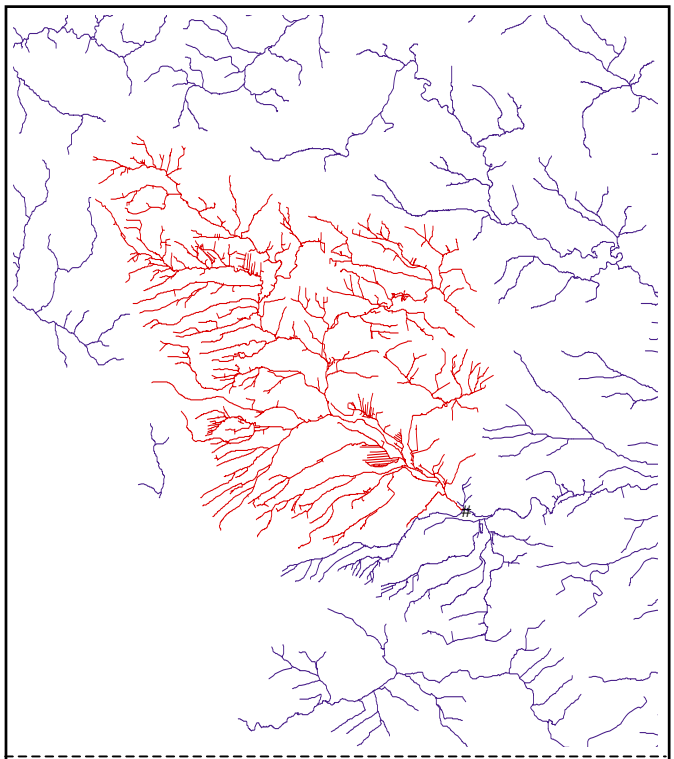


Abb. 57: Subtree über Knoten (Zusammenfluss) im Gewässernetz der UNSTRUT

Das wichtigste Strukturelement eines Baumes bildet der Knoten. Er hat im allgemeinen mindestens einen Nachfolger (engl. Child), zu dem er wiederum Vorgänger (oder Unterlieger, engl. Parent, vgl. **Kap. 4.2.2**) ist. Die Zahl der Nachfolger, die jeder Knoten haben kann, legt dessen Rang oder Ordnung fest. Ein Knoten der untersten Stufe wird auch als Blatt (bzw. im Gewässernetz als Quelle) bezeichnet. Dieser Knoten hat dann keinen Nachfolger bzw. nur „leere“ Nachfolger. Der Knoten, der keinen Vorgänger hat, wird Wurzel (Gebietsauslass) genannt. Sofern ein Knoten also nicht Blatt oder Wurzel ist, kann jeder Knoten als Wurzel eines Teilbaumes (engl. Subtree) betrachtet werden (**Abb. 57**). Die Menge der Knoten, die nicht gleichzeitig Blätter sind, wird als innere Knoten bezeichnet (im Gewässernetz sind dies die Zusammenflüsse). Ein weiteres wichtiges Charakteristikum eines Baumes ist seine Höhe, die über den maximalen Abstand eines Blattes zu seiner Wurzel definiert ist. Ausgehend von diesem Abstandsbegriff kann jedem Knoten eine Distanz zur Wurzel zugeordnet werden. Danach können verschiedene Knoten den

gleichen Abstand zur Wurzel haben. Diese Knoten werden zu Ebenen oder Niveaus zusammengefasst, die wiederum ausgehend von der Wurzel (die den Abstand 0 zu sich selbst hat) von 1 beginnend ganzzahlig bis zum entferntesten Blatt bzw. Knoten durchnummeriert werden. Diese Ebenen entsprechen den in **Kap. 4.2.2** eingeführten Hierarchieebenen oder Generationen (vgl. hierzu z. B. [DIJKSTRA 1959], [AHUJA ET AL. 1993], [JING ET AL. 1998], [KARACAPILIDIS ET AL. 1997]).

Diese Eigenschaften eines (Gewässer-) Baumes dienen vornehmlich dem Speichern und der Suche von Informationen. Als Beispiel dafür können der mittlere und der linke Baum aus **Abb. 56** herangezogen werden. Die Anzahl der Knoten ist bei beiden Bäumen gleich, die Zahl der Ebenen jedoch bei b um eins größer. Dadurch erhöht sich unter Umständen die Zeit, in der man einen bestimmten Knoten erreichen kann. In den folgenden Abschnitten werden Untersuchungen beschrieben, inwiefern diese Zusammenhänge für die Fragestellung der Arbeit genutzt werden können.

Die allgemeinste Darstellung einer Hierarchie, die in diesem Fall nicht nur auf Baumstrukturen beschränkt ist, sondern einen beliebigen Graphen darstellen kann (**Abb. 54**, **Abb. 55**), ist das Ablegen der Kanten in einer Tabelle, wobei die Tupel (V, N) die Vorgänger- und Nachfolgeknoten enthalten. Dabei müssen die Knoten eindeutig identifizierbar sein. Mit dieser Methode können beliebig geartete Hierarchien (z. B. komplexe Hierarchien, in denen Knoten mehrere Vorgänger besitzen können) abgebildet werden. Grundsätzlich kann damit jeder Graph, auch eine Menge von Graphen, in einer einzigen Tabelle gespeichert werden.

4.2.5.2 Generischer Verzweigungscode als Grundlage der Hierarchisierung

„Complex systems may be thought of as having different levels, and these levels are generally referred to as being a hierarchy“ [LOBUE 2002]. Ein Kartenwerk, der Aufbau eines Organismus, eines Betriebes, einer Maschine oder eines Gewässernetzes kann unter hierarchischen Gesichtspunkten betrachtet werden. *„It also makes clear that a complex system is really a system of systems, such that there are both: interactions between the parts, or agents of systems on a given level, and interactions across different hierarchical levels“* [LOBUE 2002]. Will man z. B. untersuchen, wie die Ökonomien Japans und Europas interagieren, ist es nötig, auch die Verhältnisse auf einem niedrigeren Level der Hierarchie – z. B. auf Betriebsebenen zu untersuchen. Hierarchische Ebenen werden von Objekten gebildet, deren Eigenschaften die Ebene bezüglich einer bestimmten Fragestellung festlegen (vgl. **Abb. 58** und **Abb. 59**). So wie eine Person unter verschieden Blickwinkeln als Arbeitnehmer, Organismus oder Tourist angesehen werden kann, kann ein Gewässerabschnitt unter dem Gesichtspunkt des Verschmutzungsgrades in einer vollkommen anderen Hierarchieebene erscheinen als bei der Betrachtung seiner räumlichen Lage.

Der Vergleich eines Organismus mit seinen nächsten und entfernteren Verwandten zeigt, dass seine Eigenschaften in hierarchischen Mustern verschachtelt (nested) sind, wodurch er seinen näheren Verwandten ähnlicher ist:

- *Vertebrata; Tetrapoda; Amniota; Mammalia; Eutheria; Homo sapiens (oder vollständiger: Eukaryotae; Metazoa; Eumetazoa; Bilateria; Coelomata; Deuterostomia; Chordata; Vertebrata; Gnathostomata; Osteichthyes; Sarcopterygii; Choanata; Tetrapoda; Amniota; Mammalia; Theria; Eutheria; Archonta; Primates; Catarrhini; Hominidae; Homo; sapiens)* [VERRAES 2001]

Der Mensch gehört wie die Maus und die Primaten zu den *Eutheria* (plazentale Säugetiere), da der Vorfahre von *Eutheria* eine Plazenta entwickelt hatte, ist jedoch den Primaten näher in der Entwicklungsstufe und damit ähnlicher. Verwandschafts- und Ähnlichkeitsgrad bestimmen sich aus der relativen Position der Elemente im Hierarchiebaum zueinander.

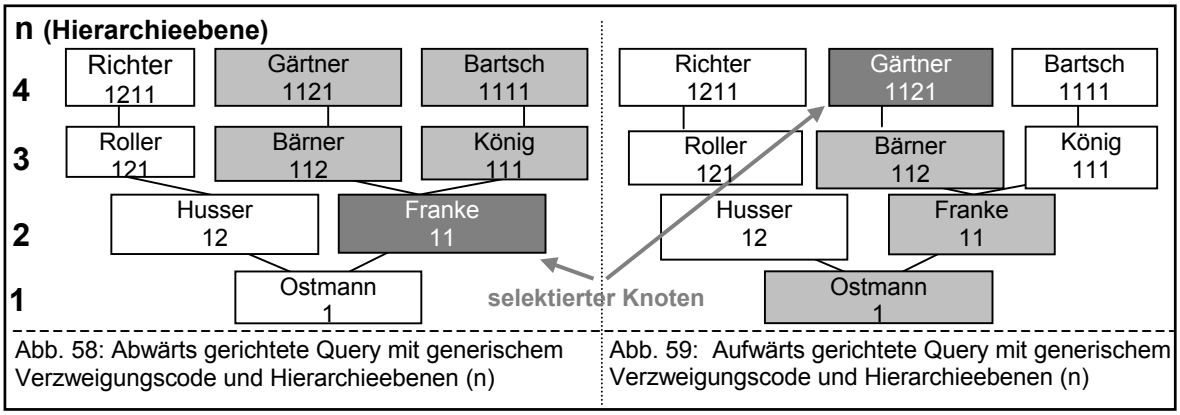


Abb. 58: Abwärts gerichtete Query mit generischem Verzweigungscode und Hierarchieebenen (n)

Abb. 59: Aufwärts gerichtete Query mit generischem Verzweigungscode und Hierarchieebenen (n)

Aus diesen Voraussetzungen ergibt sich die Möglichkeit der Abbildung von Hierarchien durch die generische Codierung ihrer Verzweigung auf Basis einer Stringverkettung. Generisch bedeutet hier, dass der Code eines Elementes auf jeder Hierarchiestufe (z. B. Betriebsleiter, Gebietsauslass) in all seinen höher liegenden Nachbarn enthalten ist, also von den Eltern (Unterlieger, Parent) an die Kinder (Oberlieger, Childs) vererbt wird. Abwärts gerichtete Querys können nach korrekter Implementierung dieses Codes relativ einfach umgesetzt werden. Wie **Abb. 58** zeigt, müsste zunächst der Code des selektierten Knotens mit den Codes der darunter liegenden Knoten verglichen werden. In diesem Beispiel wäre das die `11`. Die gesuchten Knoten enthalten diesen Code in den ersten Stellen. Die **`11`** bildet sich also in der Folge **`11`**`2, `**`11`**`1, `**`11`**`11 und `**`11`**`21` ab; in ein SQL-Statement umgesetzt würde diese Fragestellung folgendermaßen aussehen:

```
SELECT * FROM employees WHERE Sidecode like '11%'
```

Im Gegensatz dazu bedarf die aufwärtsgerichtete Abfrage (vgl. **Abb. 59**) einer Stringbehandlung, bevor das SQL-Statement formuliert werden kann. Die hier hervorgehobene Kette beginnt mit dem Code `1121`. Die folgenden Codes bis zur Basis der Hierarchie lauten `112, 11, 1`. Die Analyse zeigt, dass diese Codes und somit die gesamte Kette im Startcode enthalten sind. Die einzelnen Glieder ergeben sich in absteigender Reihenfolge aus den der Hierarchiestellung n entsprechenden ersten n Stellen des Startcodes, die durch eine einfache Schleife fragmentiert werden können. Die SQL-Abfrage für **Abb. 59** würde demnach lauten:

```
SELECT * FROM Employees WHERE Sidecode IN ('1121', '112', '11', '1')
```

Bei der Codierung werden Pseudoknoten optional nicht berücksichtigt. Dieses Übergehen von Abschnittsunterteilungen zwischen zwei Zusammenflüssen reduziert die Anzahl der für die Generationenfolge benötigten Stellen, wodurch sich die Stringlänge und somit auch der Speicherplatzbedarf verringern. Sollen die Pseudoknoten einbezogen werden, ist der benötigte Speicherplatz natürlich höher. Der beispielsweise unter ARCVIEW mögliche Speicherplatz für ein Stringfeld (hier als „SIDECODE“ bezeichnet) beträgt 255 Zeichen. Dies entspricht einer hierarchischen Distanz von 253 dazwischen liegenden Hierarchieebenen (n = 253), die bei Vernachlässigung der Pseudoknoten der Anzahl von Verzweigungen auf dem Weg zur Wurzel entspricht. Sollte diese bei sehr komplexen Gewässer-

netzen überschritten werden, so wird der Inhalt überzähliger Zeichen in ein weiteres Feld geschrieben. Dieses wird ebenfalls bis zu 255 Zeichen aufgefüllt usw., bis alle Abschnitte bearbeitet sind. Bei z. B. zehn SIDECODE-Feldern wäre also eine hierarchische Distanz von 2530 möglich. Der Größe eines auf diese Art zu hierarchisierenden Fließgewässernetzes wäre somit (abgesehen von der Speicherkapazität) keine Grenzen gesetzt.

Der Algorithmus bewegt sich nach dem Prinzip von **Code 6** sukzessiv den Hierarchiebaum nach unten. Entsprechend können sowohl zusammenhängende als auch verteilte Flussnetze (mehrere Cluster) bearbeitet werden. Als Vorgabe wird ein einzelner Abschnitt erwartet, der den Beginn der Hierarchisierung kennzeichnet. In der Regel ist dies der Auslass des Gewässernetzes. Der Startabschnitt wird einem Array hinzugefügt, welches nach dem ersten Durchlauf die darüber liegenden Abschnitte (Kinder) enthält, nach dem zweiten Durchlauf die nächste Hierarchieebene etc. Die interne Richtung der pro Durchlauf gefundenen Abschnitte wird überprüft und nötigenfalls zum Start-Abschnitt hin neu orientiert. Die Hierarchieebenen sind jeweils durch eine „Generation“ mit verschiedenen „Individuen“ gleicher Codelänge, aber unterschiedlicher Codezusammensetzung gekennzeichnet (z. B. Ebene 3 – `112, 121, 231,` ...). Das ermöglicht beispielsweise die Selektion aller Abschnitte einer Hierarchieebene. Auch hier werden die Codes der „älteren Generation“ aus dem vorhergehenden Durchlauf übernommen.

Will man die Ausrichtung der Verzweigung im Code mitführen, so muss hierfür die räumliche Anordnung jedes Mitgliedes der angeschlossenen nächsten Generation analysiert werden. Um diese Frage zu beantworten, wurde zunächst überlegt, wie sich ein einheitlicher Bezugsrahmen herstellen lässt. **Abb. 60** illustriert den Lösungsweg: Die Verlängerung des Vaterabschnittes bildet einen Winkel mit den beiden Kindern. Diese Winkel werden geordnet und zugewiesen. Die dabei ermittelte Reihenfolge an jeder Bifurkation bestimmt die Anordnung innerhalb der Generation. Dabei wird die Gültigkeit gewährleistet, indem ein Zehntel der kürzesten Distanz zum ersten Vertice aller anschließenden Linien als Radius des Kreises ($\alpha = (b*180°)/(\Pi*r)$)definiert wird.

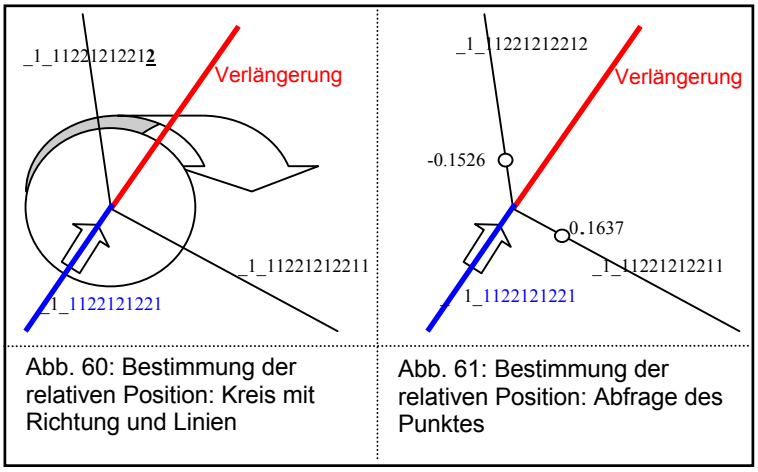


Abb. 60: Bestimmung der relativen Position: Kreis mit Richtung und Linien

Abb. 61: Bestimmung der relativen Position: Abfrage des Punktes

Eine Alternative besteht darin, diesen Radius als Position von Punkten entlang der neuen Generation zu fixieren und anschließend die relative Position dieser Punkte zur Verlängerung der Vaterlinie zu ermitteln. Das Ergebnis ist eine reelle Zahl. Ist sie positiv, so befindet sich der Punkt rechts der Verlängerung. Der kleinste positive Wert gehört also dem Punkt (und der zugehörigen Linie), der am nächsten rechts der Verlängerung liegt. Der kleinste negative Wert bestimmt die letzte Linie dieser Generation (**Abb. 61**), auf die schon in **Code 6** zugegriffen wurde.

Einschätzung:

Für hierarchische Analysen auf Basis des Generischen Verzweigungscodes ist es nicht notwendig, das gesamte zusammenhängende Flussnetz zu untersuchen. Alle Abfragen sind auch auf Teildatensätze anwendbar. Nicht zuletzt werden damit auch die Prozesszeiten verkürzt und die Übersichtlichkeit der Ergebnisse erhöht.

Die hohe Flexibilität dieser Codierung ergibt sich aus der relativ einfach zu erzeugenden SQL-Abfrage (sowohl gewässerauf- als auch gewässerabwärts). Diese kann entweder nur im RDBMS bzw. nur im GIS oder in der Kopplung beider Komponenten geschehen. Dabei gilt die Regel: erst SQL-Abfrage an die Datenbank, dann Selektion in der Karte:

```
SELECT * FROM River WHERE Sidecode like '11%' // SideCode like '11% würde dann also im GIS verwendet.
```

Alle Berechnungen/Analysen können mithilfe dieses Attributes durchgeführt werden, unabhängig von exklusiven GIS-Routinen wie etwa Topologie- oder Netzwerkfunktionen.

Der Code ist intuitiv und die SQL-Statements können ohne Schwierigkeiten vom Bearbeiter durch Analyse des Startcodes geschrieben werden.

Verschiedene Cluster lassen sich direkt in den Code implementieren (`_1_1212` vs. `43_1212`, die Ziffern vor dem Unterstrich sind identisch mit der Nummer des Clusters).

Das Flusskodierungssystem enthält aufgrund seiner Struktur eine Vielzahl von Informationen. Fragen nach zusammenhängenden Einzugsgebieten, nach Flüssen einer bestimmten Ordnung, nach flussauf- oder flussabwärts gelegenen Verbindungen sind direkt zu beantworten. Es müssen für diese Analysen keine zusätzlichen topologischen Informationen aus anderen Datenquellen und/oder Tabellen integriert werden. Dies erhöht die Unabhängigkeit und Prozessgeschwindigkeit.

Es hat sich gezeigt, das der Code besonders leistungsfähig in der Übertragung auf alternative Darstellungsvarianten der Gewässerhierarchie ist. So lässt er sich relativ einfach als Treeview abbilden (**Kap. 4.3.1** und **Kap. 4.3.2**) oder zur tabellenbasierten Visualisierung in HTML nutzen (Vgl .**Kap. 4.3.2**).

Der Sidecode kann beliebig erweitert und ergänzt werden. Wenn ein neuer Fluss innerhalb zweier Verzweigungen eingefügt wird, bedeutet dies nur die Übertragung des dort befindlichen Codes auf die neuen Abschnitte. Werden jedoch größere Bereiche eingesetzt, kann eine partielle Hierarchisierung von einem bestimmten Punkt an automatisiert durchgeführt werden. Dabei werden die Attribute der Startlinie als Basis („Vater“) der Hierarchisierung übernommen.

Allerdings können die Varianten der aufwärtsgerichteten Abfrage unter MAPOBJECTS bei großen Gewässernetzen zu langen (exponentiell ansteigenden) Prozesszeiten führen, da hier jedes Element der IN-Klausel (...`WHERE Sidecode IN ('_1_1222', '_1_122', '_1_12','_1_1', '_1_' )...`) separat untersucht wird (das trifft auch für die OR-Klausel zu).

Der wesentlichste und dabei offensichtlichste Nachteil dieser Codierung liegt jedoch in der Redundanz, die seinem generischen Aufbau zugrunde liegt. Da jedes Element all seine Vorgänger in der Hierarchie mit sich führt, wird die Verschlüsselung entsprechend lang und damit scheinbar unübersichtlich oder sogar „abschreckend“. Obwohl dieser Kritikpunkt unter den Möglichkeiten heute verfügbarer Hard- und Software sicherlich keine allzu große Rolle mehr spielen dürfte, ist er dennoch nicht zu übersehen und führte zu einer weiteren Auseinandersetzung mit dieser Problematik, deren Ergebnisse im folgenden Kapitel beschrieben werden sollen.

4.2.5.3 Verschachtelte Hierarchien („Nested Sets“)

Genau genommen geht schon das unter **Kap. 4.2.2** beschriebene System der Unter- Oberlieger-Beziehungen auf die Graphentheorie zurück („Modell angrenzender Listen“, „Adjacency List Model“, vgl. [VOLKMANN 1996]). Die Nachteile wurden dort schon diskutiert. So unzulänglich dieses Modell für die hiesige Fragestellung ist, hat es den anderen beschriebenen Ansätzen gegenüber jedoch einen entscheidenden Vorteil: aufgrund der Verwendung zweier aneinander grenzender Listen (Felder in der Tabelle) wird die Speicherung der Hierarchie mit rein numerischen Daten möglich [CHAWATHE 1994]. Der den folgenden Ausführungen zugrundeliegende Ansatz nutzt dieses Verhalten, verwirft jedoch den Versuch, die hierarchischen Beziehungen unmittelbar durch die direkten Nachbarschaften abzubilden. Die Kombination mit den Ausführungen zu Beginn von **Kap. 4.2.5.2** führte zum Modell verschachtelter Hierarchien durch eine Umwandlung des dort beschriebenen Generischen Verzweigungscodes in eine „Nested Set“–Codierung (**Abb. 62**)[30].

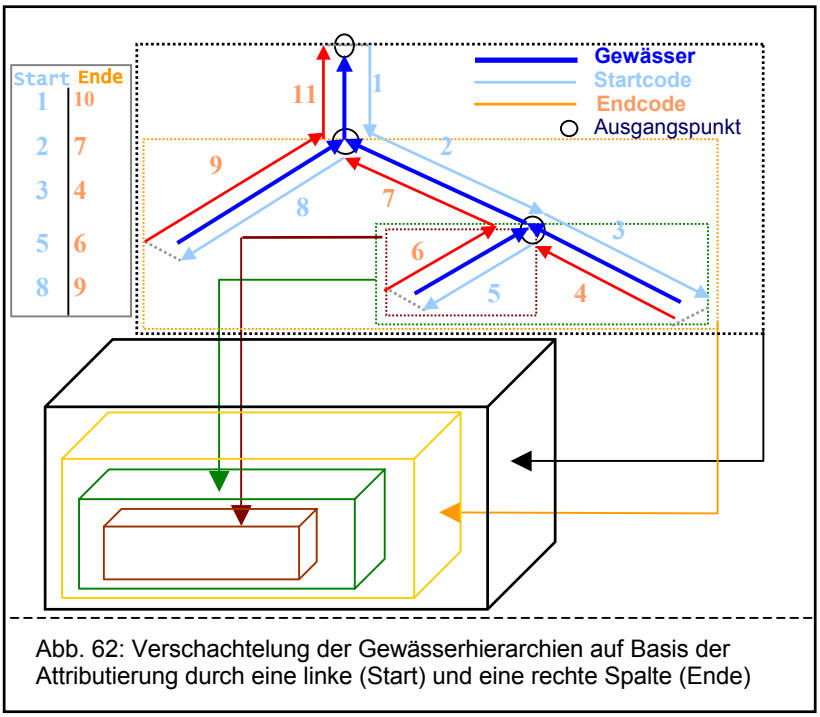


Abb. 62: Verschachtelung der Gewässerhierarchien auf Basis der Attributierung durch eine linke (Start) und eine rechte Spalte (Ende)

Nimmt man den Begriff der Verschachtelung (Nesting) wörtlich, so kann man sich vorstellen, dass jedem Abschnitt eine Schachtel zugeordnet wird, deren Größe von der Anzahl seiner Oberlieger abhängt. Alle darüber liegenden Abschnitte sind der Hierarchie entsprechend mit ihrer Schachtel dort hinein projiziert (**Abb. 62**). Um diese Idee in eine korrekte Hierarchisierung umzusetzen, stelle man sich die Grenzen jeder Schachtel als Umfassende ihrer oberhalb liegenden Abschnitte vor. Die Wurzel des Gewässerbaumes ist der Ausgangspunkt der größten Umfassenden. Hier beginnt die Hierarchisierung über zwei Attribute: START und ENDE. Die Punkte in **Abb. 62** können so auch als Pflöcke interpretiert werden, an denen der Start der Umfassungslinien verankert ist. Von dort ausgehend wird ein Counter für jede Belegung inkrementiert und, wenn der Abschnitt noch nicht durchlaufen wurde, dem Feld START zugewiesen. Die Bewegung erfolgt entweder immer rechts- oder immer links gerichtet. An einer Verzweigung bewegt sich der Algorithmus also immer im oder immer entgegengesetzt dem Uhrzeigersinn. Um dabei konsistent zu bleiben, wird die unter **Kap. 4.2.5.2** beschriebene Funktion zum Auffinden des jeweils rechtesten bzw. linksten unterhalb liegenden Nachbarn genutzt. Die Blätter des

[30] [CHAWATHE 1994] beschreibt die Abbildung verschachtelter Hierarchien in Objektorientierten Datenbanken.

Baumes (Quellen) bilden die innersten Schachteln. Werden sie erreicht, beginnt die Belegung des Feldes ENDE und damit die Vervollständigung der Attributierung (**Abb. 63**)[31].

Sichtbar wird damit auch, dass dies die natürlichste der vorgestellten Arten ist, verschachtelte Hierarchien von Gewässernetzen abzubilden. Das Verfahren hat somit einige vorhersagbare Effekte:

- Die Wurzel liefert: START = 1; ENDE = 2*(SELECT COUNT(*) FROM river).
- Quellen werden mit Start; ENDE = START + 1 zurückgegeben.
- Unterbäume sind durch die BETWEEN[32]-Klausel definiert.

Von den zwei belegten Attributen START und ENDE können nun verschiedene hierarchische Abfragen ausgehen:

- Selektiere die Oberlieger eines bestimmten Gewässerabschnittes, unabhängig davon, wie tief das hierarchische System ist:

```
SELECT *
FROM Gewaesser AS G1
WHERE G1.START BETWEEN s AND e;  //s und e sind die Werte von Start und Ende am selektierten Knoten
```

- Selektiere die Unterlieger eines bestimmten Gewässerabschnittes, unabhängig davon, wie tief das hierarchische System ist:

```
SELECT *
FROM Gewaesser
WHERE START<=s AND START>=rs AND ENDE>=e AND ENDE<=re  //s und e sind die Werte von Start und Ende am selektierten Knoten; rs und re die der Wurzel
```

- Auch die Verbindung entlang dem Gewässer kann nur durch reines SQL und ohne Stringoperationen (wenn auch etwas komplexer) erfragt werden:

```
SELECT *
FROM Gewaesser
WHERE links<=s1 AND START>=rs AND ENDE>=e1 AND ENDE<=re OR (links<=s2
AND START>=rs AND ENDE>=e2 AND ENDE<=re) XOR (START<s1 AND ENDE>e2)// die XOR –Bedingung ist der Filter
```

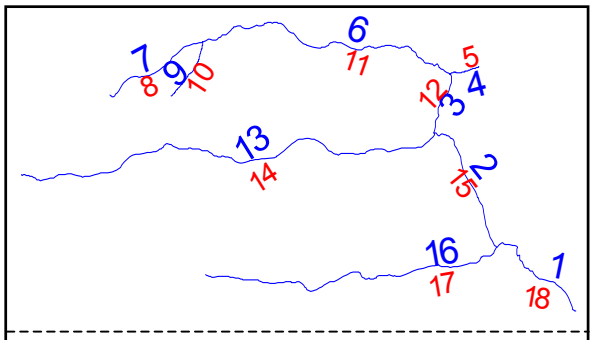


Abb. 63: Hierarchisierung als nested Set (VIEZBACH – GOLDGRUNDBACH)

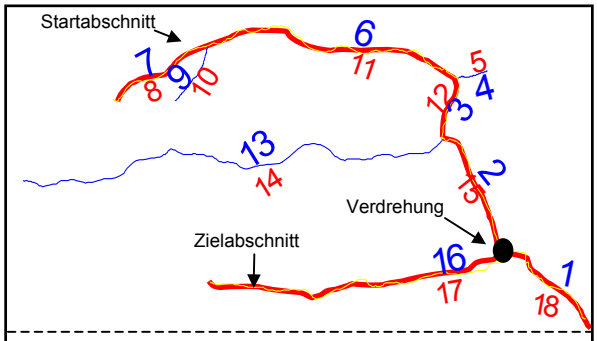


Abb. 64: Verbindung entlang dem Gewässer (VIEZBACH – GOLDGRUNDBACH)

Bei der letzten Abfrage liegt wieder eine Kombination aus zwei abwärtsgerichteten Querys und einem Filter vor. Diesen Filter erhält man, indem die beiden selektierten Stränge (**Abb. 64**) an dem Punkt gedanklich um 180° verdreht werden, wo sie sich berühren; START wird zu ENDE und umgekehrt. Dabei werden die auszuschließenden Abschnitte

[31] Man kann diese Beschreibung auch in das mentale Modell eines baumstrukturierten Gewässernetzes mit beidseitigem Radwegen transformieren. Es bestehen genau zwei Möglichkeiten, diesen Baum abzufahren: rechts und links herum. Nach jeweils 100 Metern wird die insgesamt zurückgelegte Strecke auf blauem Papier im Randstreifen abgelegt. Immer wenn ein Gewässerabschnitt schon einmal (von der anderen Seite) befahren wurde, wird der Counter auf dieser Seite mit rotem Papier fixiert. Hat man den Ausgangspunkt wieder erreicht, so würde von oben betrachtet das Schema aus **Abb. 63** sichtbar werden.

[32] Der BETWEEN – AND -Operator in SQL selektiert die Datensätze innerhalb eines bestimmten Wertebereiches.

sichtbar: es sind diejenigen, die in START den Wert des Ausgangsabschnittes und in ENDE den des Endabschnittes haben. Konkretisiert für dieses Beispiel würde die Abfrage folgendermaßen lauten:

```
SELECT * FROM Gewaesser
WHERE links<=7 AND START>=1 AND ENDE>=8 AND ENDE<=18 OR (START<=16 AND START>=1 AND ENDE>=17
AND ENDE<=18) XOR (START<7 AND ENDE>17)//die XOR –Bedingung ist der Filter
```

Obwohl der Algorithmus (**Code 22**) zur hierarchischen Attributierung anders aufgebaut und komplexer ist als beispielsweise **Code 6**, arbeitet er sehr robust und effizient auch bei großen Gewässernetzen, sofern diese zuvor nach den Maßgaben unter Kap. **4.1** kontrolliert und ggf. korrigiert wurden.

Einschätzung:

Diese Codierung weist alle Vorteile des generischen Verzweigungscodes (vgl. **Kap. 4.2.5.2**) sowie des Modells angrenzender Listen (vgl. **Kap. 4.2.2**) auf und eliminiert deren Nachteile (vgl. dazu die Übersicht in **Tab. 12** und **Tab. 13**).

Alle Abfragen sind auch hier wieder auf Teildatensätze anwendbar.

Es traten keine Redundanzen der Metadaten mehr auf. Das Modell ist somit vollständig normalisierbar.

Aufgrund der numerischen Codierung ist eine beliebige Verfeinerung des Gewässerbaumes denkbar (**Abb. 65**).

Die für die Codierung benötigte Speicherkapazität ist sehr gering. Es werden zwei Felder angelegt, die in der Lage sind, positive Zahlen von der Größe Anzahl der Gewässerabschnitte n * 2 aufzunehmen. Allerdings sollte berücksichtigt werden, dass für eine nachträgliche Verfeinerung (**Abb. 65**) die definierten Felder Fließkommazahlen müssen.

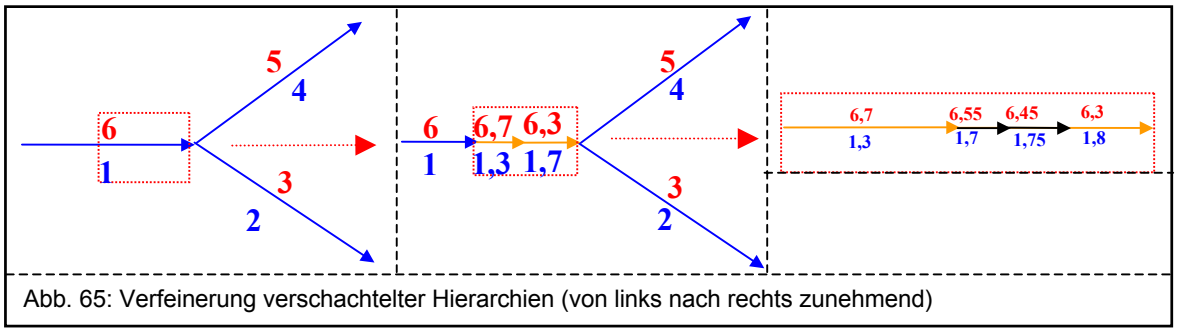


Abb. 65: Verfeinerung verschachtelter Hierarchien (von links nach rechts zunehmend)

Es ist das einzige Verfahren, mit dessen Hilfe alle drei SQL-Abfragen direkt aus der Tabelle extrahiert werden können.

Da die Codierung den Gewässerbaum zuerst hinauf- und danach hinabläuft, kann sie auch sehr gut für die alternative Abbildung z. B. in Form eines Treeviews (vgl. u.a. **Kap. 4.3.1**) oder einer HTML-Tabelle verwendet werden.

Diese Eigenschaften sind ausgesprochen gut geeignet, auch sehr große Gewässernetze (> 100.000 Abschnitte) abzubilden. Untersuchungen zu Prozessgeschwindigkeiten (vgl. **Kap. 4.2.2**) ergaben die besten Ergebnisse im Vergleich mit den anderen Systemen. Beispielsweise wurde hier der „IN“- bzw. „OR“-Bereich der SQL-Klausel bei der abwärtsgerichteten Query durch die Abfrage eines Wertebereiches ersetzt. Besonders unter MAPOBJECTS ergaben sich dadurch erhebliche Geschwindigkeitsvorteile.

Das System ist nach seiner Berechnung vom GIS unabhängig, kann jedoch auch in diesem genutzt werden. Dabei wird das SQL-Statement auf den Inhalt der `WHERE`-Klausel reduziert:

```
START<=s AND START>=rs AND ENDE>=e AND ENDE<=re
```

oder z. B.:

```
links<=7 AND START>=1 AND ENDE>=8 AND ENDE<=18 OR (links<=16 AND START>=1 AND ENDE>=17 AND ENDE<=18) XOR (START<7 AND ENDE>17)
```

Insgesamt ist das Modell verschachtelter Hierarchien von den in der Arbeit untersuchten und entwickelten Ansätzen am besten geeignet, die in **Kap. 1.2** und **Tab. 1** dargestellten Prämissen zu erfüllen.

4.2.6 Zusammenfassung und Bewertung

Die unter Kap. **4.2** vorgestellten Systeme sind für verschiedene Bereiche unterschiedlich gut einsetzbar. Ziel war es, eine möglichst optimale Eigenschaftskonstellation der Codierung zu erreichen. Dabei sind von den im Rahmen dieser Arbeit entwickelten Methoden zur Hierarchisierung von Fließgewässernetzen besonders der GENERISCHE VERZWEIGUNGSCODE und die VERSCHACHTELTEN HIERARCHIEN (NESTED SETS) hervorzuheben. Letzteres Modell weist die günstigste Eigenschaftskonstellation auf und wurde bei der Einbindung in das noch zu beschreibende hierarchiebasierte Fließgewässer-Managementsystem (**Kap. 4.3.1**) und ein internetbasiertes Fließgewässer-Informationssystem (**Kap. 4.3.2**) besonders berücksichtigt. Hierbei kann jedoch auch der generische Verzweigungscode gute Dienste leisten, da er sich ebenfalls sehr gut für die alternative Abbildung des Gewässerbaumes eignet (vgl. z. B. **Kap. 4.3.1**)

Diese beiden Flusskodierungssysteme enthalten aufgrund ihrer Struktur und Anlage die Möglichkeit der Koppelung mit einer Vielzahl von Informationen. Fragen nach zusammenhängenden Einzugsgebieten, nach flussauf- und flussabwärts gelegenen Verbindungen, nach der Flusslänge, Fließrichtung usw. sind direkt in Kombination mit den Metadaten ableitbar. In Hinblick auf die Optimierung von Einzugsgebietsmanagement und -monitoring wurden verschiedene Verfahren zur räumlichen und inhaltlichen Erweiterung der verfügbaren Informationen entwickelt, auf die in den folgenden Abschnitten eingegangen werden soll.

Einen Überblick zu den besprochenen Ordnungssystemen und ihrer Eignung in Hinblick auf die der Arbeit zugrunde liegenden Kriterien geben **Tab. 12** und **Tab. 13**.

Kriterium	Beschreibung
1	automatisiert aus der 2-dimensionalen Geometrie extrahierbar
2	Eignung für aufwärtsgerichtete Abfragen
3	Eignung für abwärtsgerichtete Abfragen
4	Eignung für verbindende Abfragen
5	iterationsoptimiert bei SQL-freier Selektion
6	effektiv in der Speicherung bzw. möglichst wenig redundant
7	erweiterbar / verfeinerbar
8	regionalisierbar / globalisierbar - es sollte zumindest als Grundlage einer globalen Ausweitung dienen können

9	intuitiv, d.h. für den Bearbeiter leicht verständlich
10	hydrologisch aussagekräftig
11	zur alternativen Darstellung geeignet

Tab. 12: Kriterien der Eignung eines Fließgewässer-Ordnungssystems zur Abbildung in DBMS und GIS

Kriterium ⇒ System ⇓	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**beschrieben und ggf. implementiert in Kapitel:**
HUC (Hydrologic Unit Code)	n	j	j	j	j	-	-	-	j	j	n	2.3
DNAEE (Departamento Nacional De Aguas E Energia Eletrica)	n	j	j	j	j	-	-	-	j	j	n	2.3
GRDC (Global Runoff Data Center)	n	j	j	j	j	-	-	j	b	j	n	2.3
PFAFSTETTER	n	j	j	j	j	-	-	j	b	j	n	2.3
UL-OL-Beziehungen	j	n	n	n	-	j	J	b	b	j	b	4.2.2
STRAHLER	j	n	n	n	j	j	b	n	j	j	n	4.2.3.1
Kl. Flussordnung/ LEHNER	j	j	j	j	j	n	J	b	b	b	n	4.2.3.2
SHREVE	j	n	n	n	j	j	b	n	j	j	n	4.2.3.3
HORTON	j	n	n	n	j	j	n	n	j	j	n	4.2.3.4
LAWA	n	j	j	j	j	n	b	b	b	j	b	4.2.4
generischer Verzweigungs-code	j	j	j	j	j	j	-	j	j	j	j	4.2.5.2
Nested Set	j	j	j	j	j	j	J	j	j	j	j	4.2.5.3
j = geeignet; n = nicht geeignet; b = bedingt geeignet; - = nicht untersucht												

Tab. 13: Einschätzung der Flussordnungssysteme nach den Kriterien aus Tab. 12

Darüber hinaus existieren zweifellos andere Schemata für diese Zwecke, da weltweit viele Organisationen mit Verantwortung im Bereich der Archivierung und des Monitorings hydrologischer Daten tätig sind. Aus der verfügbaren Literatur war jedoch kein System zu erkennen, das den Anforderungen der Aufgabenstellung dieser Arbeit im Umfang der dargestellten graphentheoretischen Ansätze gerecht wird.

4.2.7 Räumliche und inhaltliche Erweiterung der Hierarchisierung

4.2.7.1 Spezifische Punkte (Quellen, Zusammenflüsse, Senken etc.)

Die Attributtabelle des betrachteten Layers erhält optional das Feld HIEPOINTS (Hierarchical Points), das spezielle Punkte speichert (vgl. **Tab. 4**, die Speicherung der anderen dort abgebildeten Punktarten ist bei einem bereinigten Gewässernetz nicht nötig, jedoch möglich):

- der Anfang der Linie ist eine Quelle (kein Nachbar an diesem Punkt)
- der Anfang der Linie ist ein Pseudoknoten (ein Nachbar an diesem Punkt)
- der Anfang der Linie ist ein Zusammenfluss (mehr als 2 Nachbarn an diesem Punkt)

Um beispielsweise alle Quelle zu erhalten, könnte ein SQL-Statement folgendermaßen aussehen können:

```
SELECT * FROM Gewaesser WHERE HIEPOINTS=1
```

Um die Quellen als Punkte zu erhalten, müsste dann nur der jeweils erste Stützpunkt der zurückgegebenen Abschnitte ermittelt werden (vgl. **Abb. 22**, **Kap. 4.3.1**).

4.2.7.2 Kumulative Werte

Die auf **Code 6** basierenden Hierarchisierungen ermöglichen die Fortschreibung eines oder mehrerer kumulativer Attribute, wie z. B. die Entfernung einer Quelle vom Gebietsauslass bzw. von der Wurzel des hierarchisierten Subtrees. Diese wird beim Durchlauf des Algorithmus für jede Generation erfasst und den (auf dem Kopf stehenden) Baum abwärts aufsummiert. Der berechnete Wert erscheint dann in einem nutzerdefinierten Feld. Diese Information kann z. B. für die Klassifikation bzw. Extraktion bestimmter Gewässerstränge von großem Nutzen sein (vgl. **Kap. 4.2.3.3**). Mit seiner Hilfe kann z. B. der längste Zweig innerhalb eines Einzugsgebietes herausgefiltert werden:

```
SELECT * FROM River
WHERE Hipoints=1 //Quelle
ORDER BY A_Dist
```

Das Ergebnis wäre eine nach der hierarchischen Entfernung vom Gebietsauslass geordnete Abfolge von Quell-Abschnitten.

Ein anderes Beispiel der Anwendung ist die flussnetzweite Bereitstellung der Hochwasserabflüsse verschiedener Jährlichkeiten, die in einem zweistufigen Prozess erfolgt:

- Zunächst werden sogenannte Hochwasserjahresserien (die Zeitreihen der jährlichen Höchstabflüsse einer Anzahl von Pegeln) zusammengestellt und einer statistischen Wahrscheinlichkeitsanalyse unterzogen. Ergebnis dieses Schrittes sind Hochwasserabflüsse der Jährlichkeit T=10 und T=50 (Hochwasserabflüsse, die statistisch im Mittel einmal in 10 bzw. 50 Jahren auftreten) an allen verwendeten Pegeln des Flussnetzes.
- Vereinfacht gesagt, werden dann im zweiten Schritt diese punktuell bekannten Hochwasserabflüsse auf das gesamte Flussnetz übertragen (Regionalisierung). Dazu werden unter anderem statistische Beziehungen zwischen der unabhängigen Variablen „kumulative Lauflänge" Lc und der abhängigen Variablen „Hochwasserabfluss der Jährlichkeit T" HQ(T) ermittelt und mithilfe der auf Basis der Hierarchisierung ermittelten kumulativen Lauflängen auf das gesamte Flussnetz übertragen. Dabei macht man sich den in der Hydrologie seit langem bekannten empirischen Befund zunutze, dass zwischen den logarithmierten Lauflängen und Hochwasserabflüssen lineare Abhängigkeiten bestehen: $Log(HQ(T)) = (aT + bT) * Log(Lc),\ T=10,50$ [MÜLLER 2001].

4.2.7.3 Sonderfälle: Wasserscheiden, Ringstrukturen, Deltabereiche, Bifurkationen

Unter **Kap. 4.1.7.6** wurde bereits ein Lösungsweg für die Erfassung und Behandlung von Ringstrukturen aufgezeigt. Mit **Code 6** wurde ein Attribut `NET` eingeführt, das alle Ringstrukturen enthält. Dadurch ist es nach der Hierarchisierung möglich, sich die Bereiche mit Ringschlüssen anzeigen zu lassen. Dieses Attribut wird gleichzeitig in das Attribut `STOP` gespiegelt, wodurch das Gewässernetz als quasi-normalhierarchisch behandelt wird und den Einstieg in eine Ringstruktur von der weiteren Bearbeitung ausschließt (vgl. **Abb. 32**).

Für den Fall, dass ein Gewässer über eine flache Wasserscheide zurückgestaut wird, können auf diese Weise auch Wasserscheiden in die Hierarchisierung implementiert werden, indem sie entweder manuell im Feld `STOP` als 1 verankert oder durch geometrische Operationen (Intersektion, vgl. **Abb. 66**) auf dieses projiziert werden. Die 1 garantiert dann den Abbruch des Algorithmus an diesen Abschnitten und gewährleistet die Integrität des Gewässernetzes im Verhältnis zum unterlagernden Relief.

Auch in Deltabereichen können bestimmte Äste über das Attribut `STOP` von der Hierarchisierung ausgeschlossen werden. Geschieht dies nicht, so würden bei einer aufwärtsgerichteten Abfrage vom Gewässerauslass aus auch diese Nebenmündungen mit erfasst werden.

Verzweigungen (echte Bifurkantionen[33]) können ebenfalls über ein `STOP`-Attribut markiert und bei der Hierarchisierung als solche berücksichtigt werden. Mithilfe eines unterlagernden DGM lassen sich diese Strukturen auch automatisiert erkennen und bearbeiten.

4.2.7.4 Hierarchisierung flächen- und punkthafter Elemente

Überlagerungen im Sinne der Verschneidung verschiedener Datenschichten können als eine erste Stufe der Analyse bezeichnet werden [BLASCHKE 1997]. Dies ist sowohl im RDBMS als auch im GIS mithilfe implementierter Funktionen bzw. Schlüsselwörter (wie AND, OR, XOR im RDBMS oder Intersection, Union, Difference im GIS, **Abb. 66**) möglich.

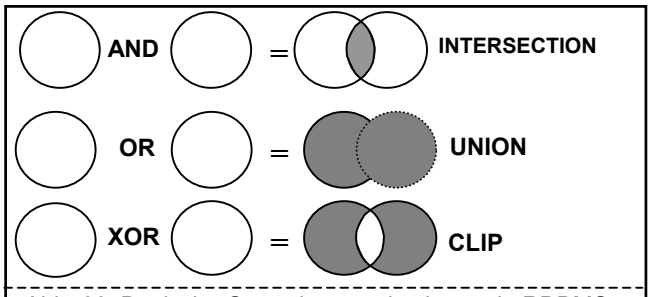


Abb. 66: Boolsche Operationen, wie sie u.a. in RDBMS und im GIS Anwendung finden

Verschneidungsoperationen verbinden nach der Booleschen Logik mindestens zwei Datensätze miteinander, etwa zur Bestimmung von Flächen mit bestimmten Attributkombinationen oder Schnittpunkten von Linien (beispielsweise Wasserscheiden mit dem Gewässernetz). Grundlage der Hierarchisierung gewässerrelevanter Rauminformationen ist der Bezug zum hierarchisierten Gewässernetz. Dieser kann

- ohne direkte räumliche Übereinstimmung und dann rein attributiv über ein Schlüsselfeld realisiert sein oder
- räumlich erfolgen. In diesem Fall kann entweder
  - eine Attributübertragung vom Gewässernetz auf das betreffende Thema oder
  - die Etablierung der Verbindung über ein Schlüsselfeld erfolgen.

[33] Ein klassisches Beispiel ist der aus dem Orinoko im südlichen Venezuela hervorgehende RIO CASIQUIARE.

Die Vorteile der Verlinkung liegen in der strengeren Einhaltung der Normalisierungsvorschriften. Diese zielen in erster Linie auf effektive, vor allem weitestgehend nichtredundante Datenhaltung. Die Verbindung via SQL erfolgt in diesem Fall durch Joins. Der Nachteil besteht in der persistenten Abhängigkeit der jeweiligen räumlichen Informationsschicht vom Gewässernetz, wodurch die Attributtabelle des Gewässernetzes in jede hierarchische Abfrage einbezogen werden muss.

Bei der Attribut-Übertragung (**Abb. 67**) ergeben sich die Nachteile aus den Vorteilen der Verlinkung: mit der physischen Einbindung der hierarchischen Attribute in die Tabelle z. B. des Teileinzugsgebietsnetzes liegen diese Informationen sowohl in dieser als auch den Attributtabellen der Fließgewässer (redundant) vor. Damit werden beide Tabellen jedoch voneinander unabhängig. Die transferierte Hierarchie kann dann mit denselben SQL-Statements (abgesehen natürlich von Tabellen- und speziellen Feldnamen) abgefragt werden.

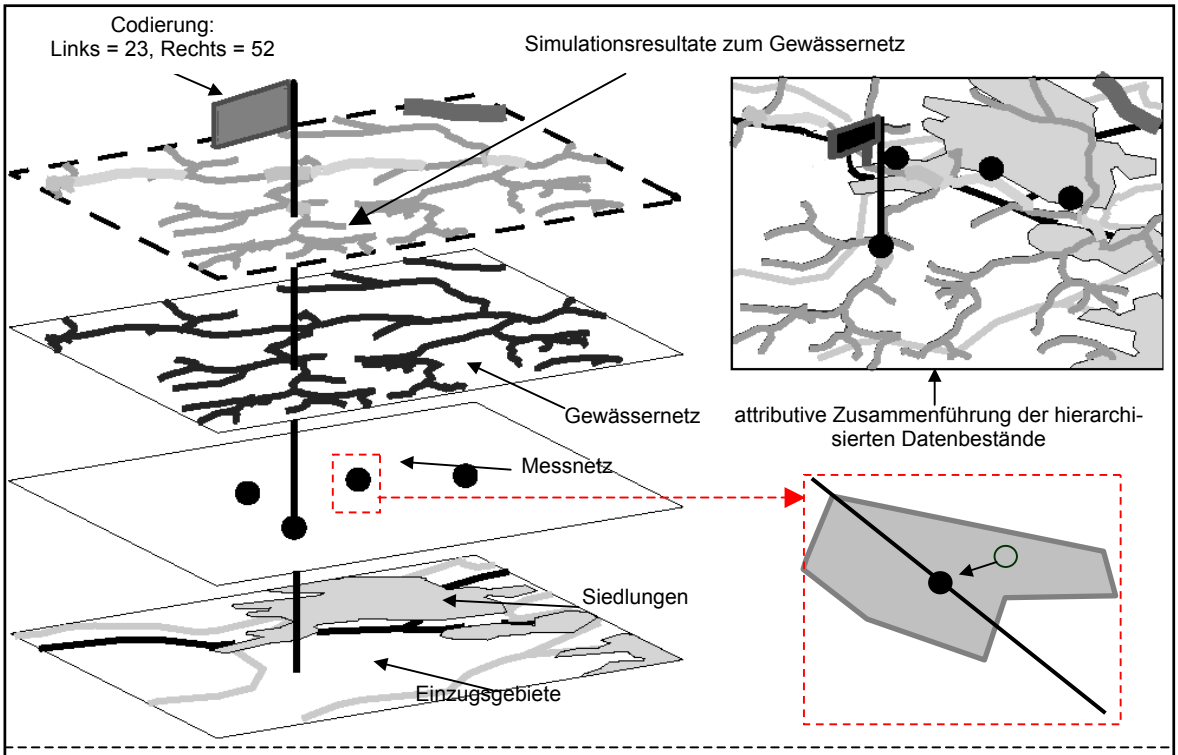


Abb. 67: Hierarchisierung verschiedener geographischer Informationsschichten im Umfeld eines Gewässernetzes durch Attributübertragung
Das Fähnchen links symbolisiert die Übertragung der Informationen der einzelnen Layer mithilfe räumlicher Abfragen.
Unten rechts: zunächst werden die Punkte eines Messnetzes – entweder virtuell oder physisch – einem Abschnitt des Gewässernetzes zugeordnet. Danach kann die Attributübertragung vom Gewässer auf die Punkte erfolgen.

Der Vorgang der Attributübertragung ist nach verschiedenen Kriterien steuerbar. **Abb. 67** schematisiert die Bedingungen. Im Falle eines Polygonthemas wird eine „Punkt – in - Polygon“ - Abfrage durchgeführt. Von allen innerhalb des Polygons liegenden Gewässerabschnitten werden die hierarchischen Metadaten einer Linie übertragen. Bei Punkten (z. B. eines Messstellennetzes) können die Attribute der jeweils nächsten Linien auf die Punkte übertragen werden oder umgekehrt von den Punkte auf die Linien. Darüber hinaus wurde eine Funktion implementiert, welche die Punkte eines Themas auf die jeweils zunächst

liegende Linie, bzw. innerhalb dieser auf das nächstliegende Vertice versetzt und an dieser Stelle trennt (**Abb. 67** rechts unten). Somit ergibt sich die Möglichkeit, hierarchische Metadaten uneingeschränkt in anderen Informationsschichten anzuwenden.

4.2.7.5 Regionalisierung / Globalisierung

Mithilfe eines anderen zusammenhangbeschreibenden Attributes (z. B. CLUSTER) können die vorgestellten Codierungssysteme global angewendet werden. Dabei lassen sich auch räumlich verteilte Systeme gemeinsam verwalten (Globalisierung) bzw. physisch zusammenhängende Gewässernetze attributbasiert in unterschiedliche Subsysteme aufteilen (Regionalisierung, vgl. **Abb. 68**). Das SQL-Statement wird dann um die Einschränkung durch dieses Attribut erweitert:

```
Select * FROM River WHERE Sidecode like '1223%' AND Cluster=432
```

oder

```
Select * FROM River WHERE ([START] >= 206) AND ([ENDE] <= 295) AND Land='Southafrica'
```

Ohne diese Einschränkung würde das Ergebnis in verteilten Flussnetzen nur in einem Cluster sinnvoll sein.

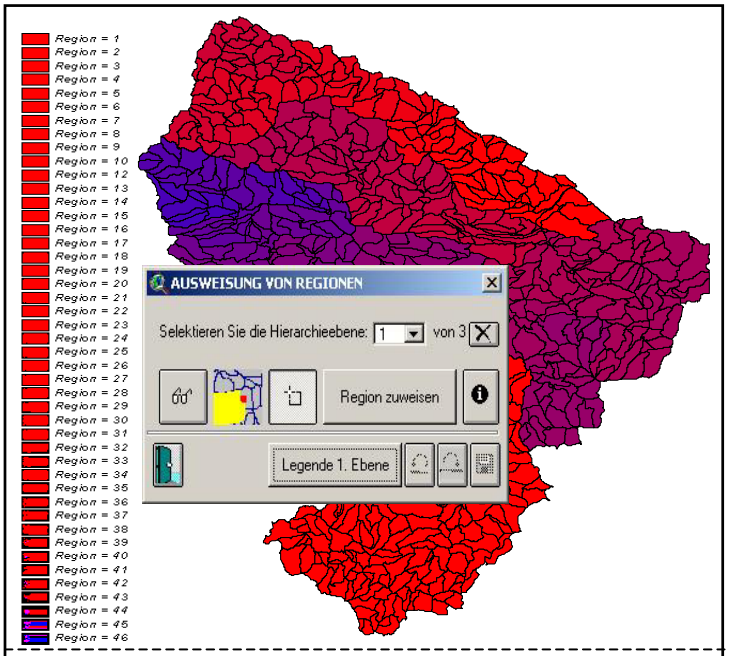


Abb. 68: Regionalisierung von Teileinzugsgebieten (UNSTRUT)

Häufig ist es notwendig, die bei der Abbildung mancher Teilprozesse benötigte hohe räumliche Auflösung für die Simulation anderer Prozesse wieder zu vereinfachen. Außerdem wird insbesondere bei der meso- bis makroskaligen Modellierung oft genestet gearbeitet, d.h. innerhalb des gesamten Untersuchungsgebietes werden Teilregionen detaillierter analysiert [HABERLANDT ET AL. 2000]. Die in **Abb. 68** dargestellte Applikation diente u.a. zur Vorbereitung der Simulation des Wasser- und Stoffhaushaltes einer sich verändernden Naturlandschaft im Nationalpark Bayerischer Wald (NP BW)[34], indem Teileinzugsgebiete zu größeren Regionen zusammengefasst wurden.

[34] Einzugsgebiete der Großen Ohe

4.2.7.6 Nicht-baumstrukturierte Netzwerke (Routing-Algorithmen)

Vernetzte Strukturen, wie sie im Zusammenhang mit Kanalbauten und besonders in Flachlandbereichen häufig anzutreffen sind, können durch die bisher vorgestellten Codierungen nur mit Einschränkungen oder manueller Unterstützung abgebildet werden. Sollen z. B. Punkte innerhalb solcher Netzwerke verbunden werden, stellt sich das sogenannte „Shortes Path - Problem“ (**Abb. 69**). Es findet auf vielen Gebieten wie z. B. der kombinatorischen Optimierung, im Transportwesen, militärischer Logistik oder der künstlichen Intelligenz Anwendung, wo die kürzeste Verbindung eines jeden Knotens zu einem bestimmten Zielknoten ermittelt werden soll [MAHESHWARI & ZEH 2001].

Der DIJKSTRA-Algorithmus [DIJKSTRA 1959][35] ist das wohl bekannteste Verfahren, wenn man mit ungerichteten Graphen arbeitet (vgl. **Kap. 4.2.5.1**). Er findet den kürzesten Weg zwischen einem Start- und einem Zielknoten[36]. Ein Pfad ist ein Weg, in dem kein Knoten mehr als einmal erscheint. Eine Kreisstruktur ist demnach ein Pfad, in dem nur der Startknoten zweimal enthalten ist. Hier die notwendigen Ausgangsbedingungen zur Ermittlung des kürzesten Weges innerhalb eines gerichteten Graphen:

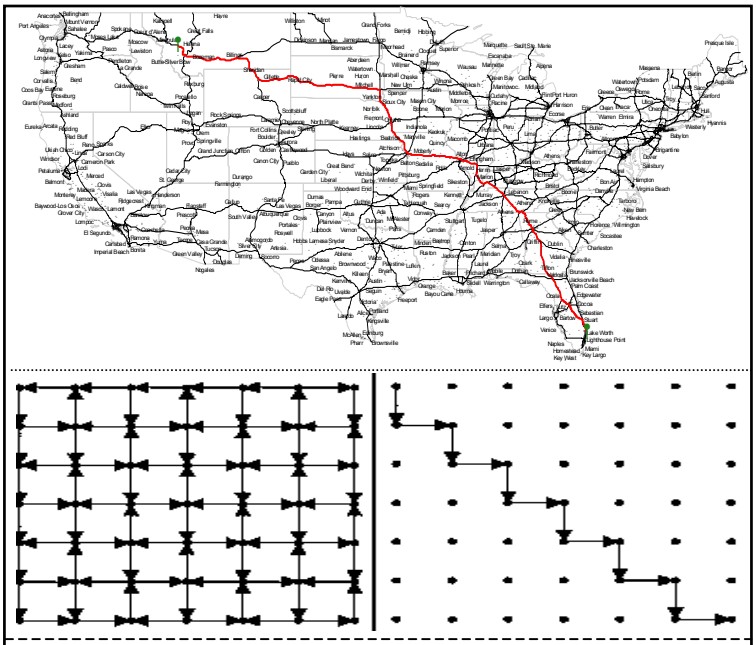


Abb. 69: Graphentheoretischer Hintergrund des DIJKSTRA-Algorithmus
oben: Strassennetz der USA mit kürzester Verbindung (Pfad)
unten: Adjazenzmatrix mit Pfad

- der Graph muss zusammenhängend sein
- die Knoten müssen separat gespeichert vorliegen
- alle Verbindungen und ihr Wert (Kosten - in der Regel die Länge) müssen mit den Knoten gespeichert sein.

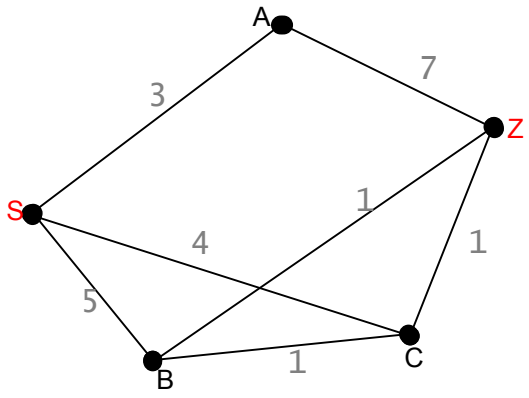
Auf Basis dieser Bedingungen wurde ein AVENUE-Script entwickelt, das diese Informationen aus der Shape-Geometrie (z. B. ein korrektes Linienthema oder die Städte eines Landes) extrahiert und als Tabelle speichert, mit der danach in anderen Systemen gearbeitet werden kann. Beispiel:

```
Array Network:
Name: Stadt1
links: 2, 15, 21, 23 //Stadt 2, 4, 8, 23
maxLinks = 4 //Maximale Anzahl an Links
costs: 20.13, 41.15, 35.8, 12.45 //Distanzen zwischen Stadt1 und den unter links angeführten direkt mit Stadt1 verbundenen
//Städten, zB. in km
```

[35] E.W. DIJKSTRA: Professor der Informatik (gegenwärtig am MIT), viele Untersuchungen zur Graphentheorie und zum systematischen Programmieren (DIJKSTRA-Controllstrukturen)
[36] DIJKSTRA hat nachgewiesen, dass man den kürzesten Weg von einem Punkt zu allen anderen in der selben Zeit finden kann (sofern es sich um einen sogenannten „Spanning Tree“ handelt, einen zusammenhängenden Graphen – also z. B. ein Cluster).

Der Algorithmus kann in zwei Schritte unterteilt werden (**Code 23**) und wurde sowohl in AVENUE als auch VISUAL BASIC umgesetzt. Im Initialisierungsschritt werden drei Listen erzeugt:

- In der `IncludedList` werden alle Knoten des Graphen aufgenommen, zu denen während des Berechnungsschritts bereits die kürzeste Distanz ermittelt wurde: `IncludedList = {}`.
- In der `CandidatesList` stehen alle Knoten, die im nächsten Schritt Kandidaten für die `IncludedList` sind: `CandidatesList = {Startknoten}`.
- In einer `DistList` werden die aktuell kürzesten Entfernungen vom Startknoten zu jedem anderen Knoten im Graphen gespeichert, zunächst jedoch – ausgenommen den Startknoten – mit „Unendlich“ initialisiert: `DistList[i] = unendlich; DistList [Startknoten] = 0`.
- In einem Berechnungsschritt (**Abb. 70**) wird dann nacheinander jeweils ein Knoten aus der `CandidatesList` in die `IncludedList` aufgenommen, bis auch der Zielknoten dort enthalten ist. Dabei muss jedoch gewährleistet sein, dass der Abstand des Knotens zum Startknoten im Vergleich zu allen anderen im nächsten Schritt erreichbaren Knoten (Knoten aus der `CandidatesList`) minimal ist. Danach werden alle seine Nachbarn daraufhin überprüft, ob ihre Distanz zum Startknoten aktualisiert werden muss. Falls die bisherige Distanz zu einem Nachbarknoten größer ist, als die Distanz zum neu in die `IncludedList` aufgenommen Knoten, so wird natürlich letztere gewählt, da dies die derzeit kürzeste Entfernung vom Startknoten zu dem Nachbarknoten ist. Der Berechnungsschritt wird solange wiederholt, bis der Zielknoten gefunden ist.



Pfad von S nach Z:

Punkt	Zielpunkt	∑ Distanz
S		0
	A	3
	B	5
	C	4
A niedrigste Distanz, deshalb A als Startpunkt		
	Z	10
	S	entfällt da bereits Ausgangspunkt
C niedrigste Distanz (A, S entfallen)		
	B	5
	Z	5
Wähle B		
	Z	6-->5 (weil zu weit)
SCZ ist der günstigste (kürzeste) Weg		

Abb. 70: Graphisches Beispiel mit Lösungsweg für den DIJKSTRA-Algorithmus: zur Visualisierung sind die Distanzen hier nicht maßstäblich dargestellt. Sie können auch als Kosten (beispielsweise Distanz + Gefälle o. ä.) bezeichnet und betrachtet werden.

Trotz der erwiesenen Nützlichkeit dieser Algorithmen, sind sie mit wachsender Größe des Graphen „schleifenlastig“, ineffizient. Dabei werden Pfad-Algorithmen wie der von DJIKSTRA als nicht-hierarchisch betrachtet. Der Graph mit all seinen Bestandteilen (Knoten und Kanten) befindet sich auf ein und derselben Ebene. Er ist planar (ausgenommen den Fall von Unter- oder Überquerungen) und hat Kanten einer bestimmten Länge. Es gibt auf diesem Gebiet Untersuchungen zur Optimierung solcher Verfahren (z. B. [CAR & FRANK 1994], [MCKINNEY 1997], [JING ET AL. 1998], [WAGNER & WILLHALM 2003]). Hier wird versucht, den Graph in themenbezogen verkleinerte und über die Knoten miteinander verbundene Einheiten aufzuspalten. Das Ergebnis ist ein hierarchischer Graph. Er besteht aus einer Anzahl von Subgraphen, die auf Basis eines bestimmten Kriteriums (z. B. Stra-

ßenarten, Reisegeschwindigkeit, landschaftliche Schönheit, Fließgeschwindigkeit) oder einer Kombination verschiedener Faktoren in Klassen bzw. Ebenen unterteilt werden. Über gemeinsame Knoten, kann man sich zwischen den Ebenen bewegen. Untersuchungen (z. B. [CAR & TAYLOR 1999]) an synthetischen Graphen haben gezeigt, dass dieses Verfahren eine signifikant schnellere Pfad-Rückgabe ermöglicht, als ein flacher Algorithmus (durchschnittlich 100% bei Graphen mit mehr als 100 Knoten), weil immer nur jeweils relevante Daten betrachtet werden. Allerdings erfolgt der Geschwindigkeitsgewinn zum Teil auf Kosten der Pfadlänge.

Dem Bearbeiter sind keine Verfahren bekannt, mit deren Hilfe diese Algorithmen durch Standard-SQL abgebildet werden können. Sie sind somit auch nicht in normalen RDBMS nutzbar[37]. Wie hier dargestellt wurde, beziehen sie sich auf komplexe Netzstrukturen. Ein Flusssystem ist dabei nur ein Sonderfall und kann und sollte gesondert behandelt werden.

4.2.7.7 Verbindungsregeln

Mit der Etablierung eines baumstrukturierten Netzwerkes ergeben sich hinsichtlich der Datenintegrität weitere Optionen. Hier können vielfältige Regeln für die Ersterfassung und Pflege definiert werden. Z. B. legen so genannte Verbindungsregeln fest, welche Kante (Gewässerachse) über welchen Knoten (z. B. Mündung, Bilanz- oder Einleitungspunkt) mit welcher anderen Kante verbunden werden kann oder wie viele Kanten an einem Knoten zusammenkommen können. Beispielsweise ist es sinnvoll zu definieren, dass ein Einleitungspunkt nicht mit einer Flussuferlinie, sondern nur mit der Gewässerachse verbunden werden kann oder ein Gewässerabschnitt maximal nur eine Senke besitzen darf, oder dass an einem Bilanzpunkt maximal nur zwei Gewässerachsen zusammentreffen dürfen (andernfalls handelt es sich um eine Mündung, die einem „erweiterten“ Bilanzpunkt entspricht). Letztendlich tragen diese Regeln zu einem konsistenten Datenbestand sowohl hinsichtlich des geometrischen als auch des logischen Netzwerkes bei. Fehlerminimierte GIS-Datenbestände sind dabei entscheidend für die Verlässlichkeit aller Aussagen (**Kap. 4.1.1**).

[37] Vorstellbar wäre jedoch eine Abbildung in Netzwerkobjekten innerhalb eines OODBMS (**Kap. 2.4.2**).

4.3 Verwendung hierarchisierter Gewässernetze

4.3.1 Anwendung in einem hierarchiebasierten Einzugsgebiets-Informations- und Monitoringsystem (HEIS)

> *"'What do you consider the largest map that would be really useful?'*
> *'About six inches to the mile.'*
> *'Only six inches!' exclaimed Mein Herr. 'We very soon got to six yards to the mile. Then we tried a hundred yards to the mile. And then came the greatest idea of all! We actually made a map on the scale of a mile to the mile!'*
> *'Have you used it much?' I enquired.*
> *'It has never been spread out yet,' said Main Herr: 'the farmers objected; they said it would cover the whole country and shut out the sunlight! So now we use the country itself as its own map.' " [Carroll 1982].*

Wasserbedarf und Wasserangebot sind von Natur aus selten in Einklang zu bringen und vor allem ersterer ist durch sich verändernde menschliche Bedürfnisse einem ständigen Wandel unterworfen[38]. Die Bewirtschaftung der Ressource Wasser bedeutet daher in aller Regel einen Eingriff in die natürlichen hydrologischen Verhältnisse. *„Sustainable water resource management is a concept that emphasizes the need to consider the long-term future as well as the present."* [IWRA 2000]. Diese Nachhaltigkeit ist nur unter Berücksichtigung und Optimierung der folgenden Faktoren möglich:

- Sicherstellung der Versorgung von Bevölkerung und Industrie mit Wasser
- Sicherstellung der jeweils geforderten Qualität
- Minimierung des Eingriffes in die Natur
- Wahrung der Sicherheit von Bevölkerung und Transportwegen, z. B. bei Hochwasser.

Diese Aufgaben befinden sich im Schnittbereich vor allem von Naturwissenschaften, Ingenieurwissenschaften, ökonomischen Wissenschaften und der Politik [MANIAK 1992]. Die Wasserwirtschaft hat in den vergangenen Jahrzehnten methodische und technische Impulse erhalten, die in hohem Maße von modernen Informationstechnologien bestimmt wurden. Durch deren Einführung im Flussgebietesmanagement sind gute Voraussetzungen zur Bewältigung alter und neuer Herausforderungen gegeben, wie sie vor allem auch aus der EU-Wasserrahmenrichtlinie resultieren (**Kap. 1**).

Interaktion und Kommunikation bei der Arbeit mit einem modernen Fliessgewässermanagementsystem finden über die Benutzeroberfläche statt, die von A.U. FRANK als "... *the system for the user ...*" [FRANK 1993, S. 18] bezeichnet wurde. Die Gestaltung der Benutzeroberfläche sowie die darin angebotenen Interaktions- und Eingabemöglichkeiten wirken somit direkt auf die Konsistenz und Verständlichkeit des gesamten Systems. Es ist grundsätzlich möglich, verschiedene Sichten auf ein Gewässernetz bereitzustellen und damit ganz unterschiedliche Funktionalitäten zu verbinden (**Abb. 71** bis **Abb. 73**).

Die primäre Sicht auf ein Gewässernetz mittels GIS ist natürlich geometrisch. Die vorangegangenen Kapitel beschäftigten sich eingehend mit diesem Aspekt. Eine solche Sicht führt über angewandte Layoutfunktionalitäten zu Karten, die dann für Informations- bzw. Berichtspflichten zur Verfügung stehen - ohne Frage eine sehr wichtige Funktionalität. Analysiert man jedoch das Gewässernetz z. B. aus dem Blickwinkel eines Verantwortlichen in der Mengenbewirtschaftung oder eines Anwenders, der sich mit hydraulischen Modellen z. B. für den Hochwasserschutz befasst, dann besteht es aus „Kanten" (Fließ-

[38] Aber auch das Wasserangebot schwankt, in unterschiedlichen räumlichen und zeitlichen Betrachtungsmaßstäben, z. B. durch Bergbau oder den globalen Wandel des Klimas.

gewässerlinien) und Knoten (z. B. Mündungen, Einleitungen, Wehren und Schleusen, **Kap. 4.2.5.1**). Unterschiedliche Anwendungs- und Betrachtungsweisen führen in der Regel zu redundant gehaltenen Daten, da ein Modell meist nicht als umfassendes Auskunfts- und Kartografiesystem dienen kann und man umgekehrt im GIS meist keine komplexen hydraulische Berechnungen durchführt.

Warum sollte jedoch ein GIS nicht auch als **Verwaltungssystem für die Netzwerksicht** dienen können? Die innerhalb der Arbeit entwickelten Methoden zur Datenaufbereitung und Hierarchisierung ergeben Möglichkeiten, die vom Standpunkt der WRRL von großem Interesse sind. Hier wird ein logisches Netzwerk vollständig ohne zusätzliche Anforderungen an den Anwender im Hintergrund aufgebaut. Die konsistente Pflege ist, wie erläutert, durch die Funktionen zum Präprozessing problemlos möglich (**Kap. 4.1.1**).

Die entwickelte Anwendung besteht aus einem System von Informationen mit räumlichem Bezug, welches ein GIS, gekoppelt an eine Datenbank umfasst. Hinsichtlich der integrierten Informationen ist die zugrunde liegende Datenbankverwaltung erweiterbar und wegen der weiten Verbreitung des Programms in MSACCESS realisiert, wurde jedoch teilweise auch im Umfeld einer räumlichen Datenbank (Oracle + SDE) implementiert.

Für den Anwender ergeben sich aus dem logischen Netzwerk einige signifikante Vorteile. Durch die Speicherung des Gewässernetzes in Form netzwerkrelevanter Topologiebeziehungen als Metadaten kennt das System zu jedem Knoten die angrenzenden Kanten und jede Kante kennt ihre begrenzenden Knoten – auch unabhängig von der Geometrie. Alle gewässerrelevanten Punktinformationen wie Quellen, Senken, Zusammenflüsse etc. (vgl. **Kap. 4.1.2**, **Kap. 4.2.7.4**) sind an das Gewässernetz gebunden. Die in den Metadaten etablierte Hierarchie ermöglicht vielseitige Funktionalitäten, zum Teil in Kombination mit den notwendigen Modellkomponenten, wie im Folgenden beschrieben.

Durch Markieren eines beliebigen Punktes auf dem Gewässernetz kann das System den **Zusammenhang der Gewässernetzes** mit diesem Punkt und somit dazugehörige Objekte wie Mündungen, Einleitungen oder Entnahmen ermitteln und farblich hervorheben (**Abb. 71**). Dies ermöglicht eine ausgezeichnete visuelle Kontrolle auch bei sehr großen Datenbeständen bezüglich „sauberer“ Topologie und richtiger Zuordnung, z. B. bei der Frage, ob die Fließgewässer den Hauptströmen und damit der Flussgebietseinheit korrekt zugeordnet sind. Mit dieser Funktion lassen sich folgende grundlegende Fragen beantworten (vgl. auch **Kap. 1.2**, **Abb. 76**):

- Welche Menge eines bestimmten Stoffes fließt (kumulativ) in einen bestimmten Abschnitt? Welche Ortschaften liegen im Einzugsgebiet dieses Abschnittes? Hier müssen zunächst alle **Oberlieger** selektiert und der Gewinn oder Verlust pro Abschnitt bzw. Station berechnet werden. Dabei lässt sich eine **Statistik** zu den Berechnungen hinzu schalten[39].
- Wie viele Zusammenflüsse und Quellen befinden sich oberhalb eines Abschnittes? Diese Information enthält zugleich die Ordnungsnummer nach **SHREVE** (**Kap. 4.2.3.3**).
- Welche **Teileinzugsgebiete** (TEZGs) liegen oberhalb eines Punktes? **Oberflächenabflusslose TEZGs** lassen sich wegen der fehlenden Deckung mit Gewässerabschnitten bei der Hierarchisierung problemlos ermitteln, da sie mit der Abfrage nach den Oberliegern nicht erfasst werden (**Abb. 74**). Ihre Ausweisung ist jedoch nur bei korrekter Digitalisierung aller beteiligten Datenschichten gültig.

[39] Um z. B. die Bilanz für den Durchfluss an einem bestimmten Abschnitt zu ermitteln, sind weitere Informationen, wie Querschnitt, Neigung und die Abflussmenge an den oberliegenden Abschnitten notwendig.

- Welche **Flussdichte** bzw. **Flusshäufigkeit** und welches **Verzweigungsverhältnis** finde ich oberhalb einer Messstelle?

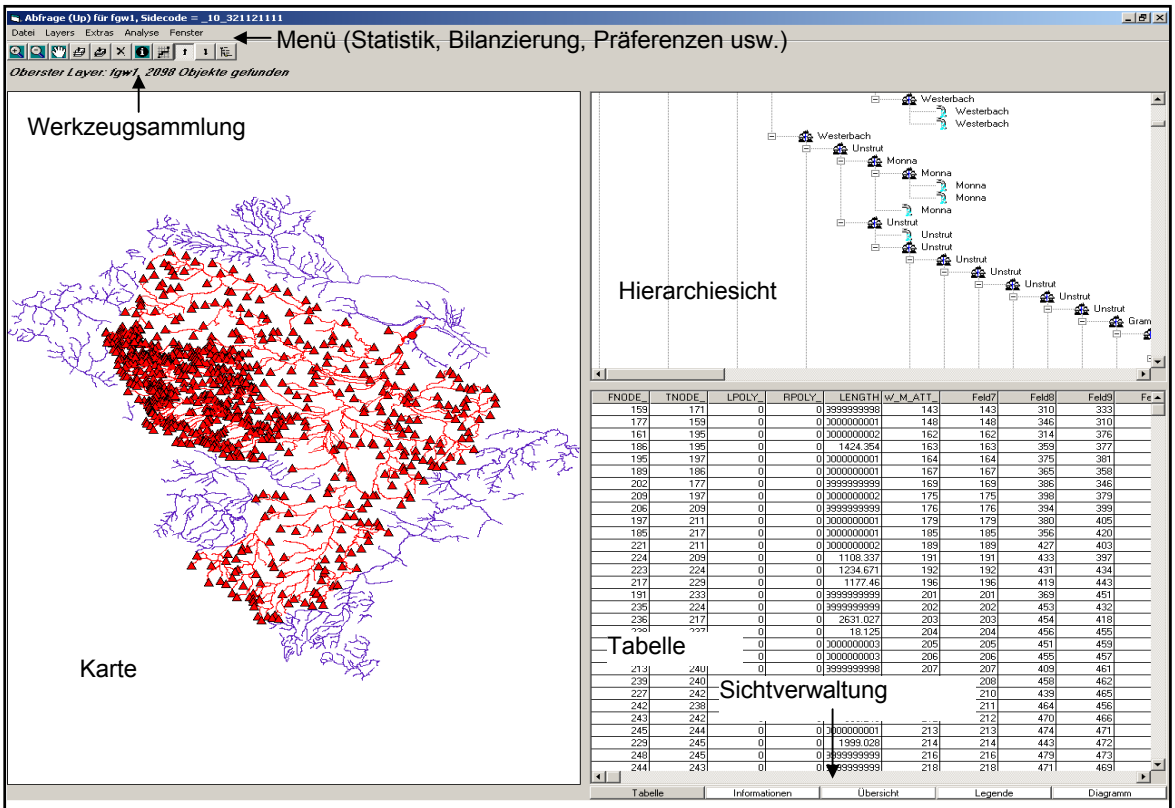


Abb. 71: Gewässerzusammenhang, Quellen, Hierarchiesicht und Tabelle auf einen Blick (Gewässernetz der UNSTRUT)

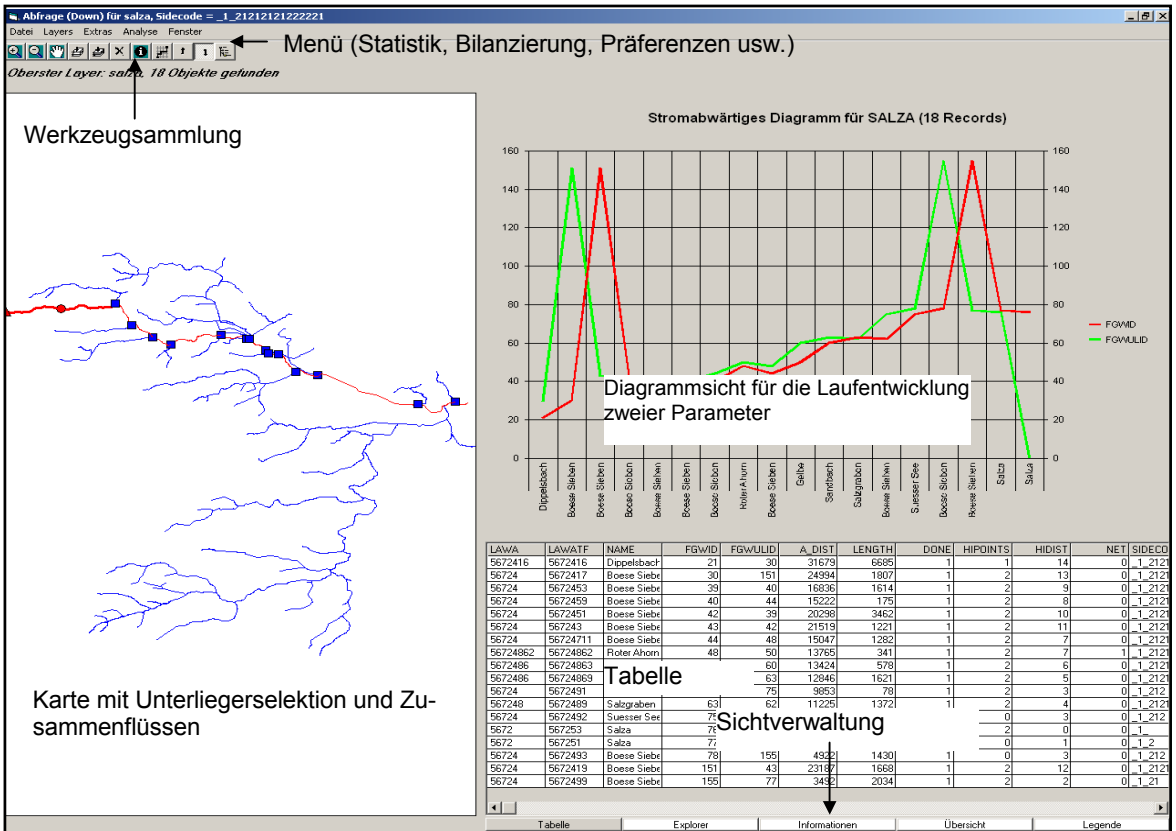


Abb. 72: Unterliegerselektion, Zusammenflüsse, Diagrammsicht und Tabelle auf einen Blick (SALZA)

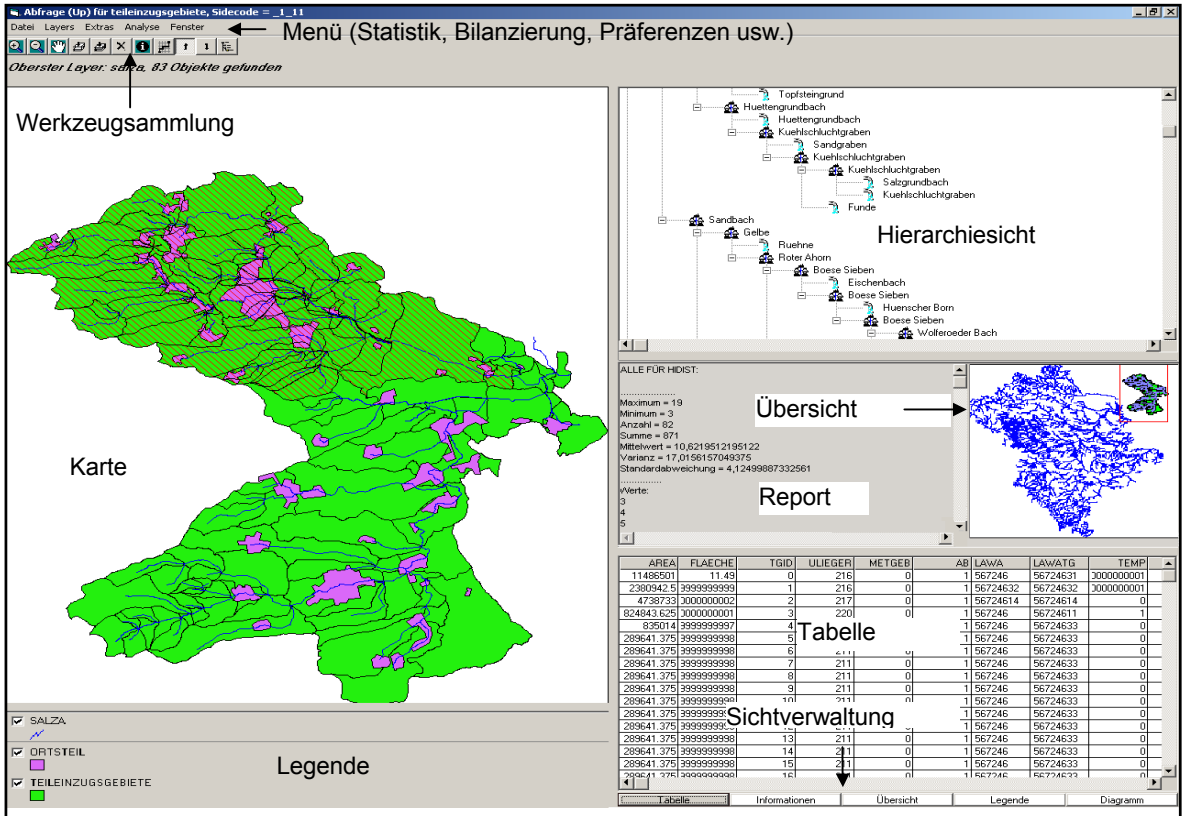


Abb. 73: Hierarchisierte Datenschichten im EZG der SALZA: Gewässernetz, Einzugsgebiete, Siedlungen; Hierarchiesicht, Report, Legende und Tabelle auf einen Blick (die schraffierten Bereiche in der Karte sind auch in der Tabelle, dem Treeview und dem Report abgebildet)

- Wie ist das **Verhältnis** von z. B. Orthophosphat zu Gesamtphosphor für alle Messstellen oberhalb eines Punktes (**Abb. 77**).
- Welche **Konzentrationsdifferenzen** ergeben sich zwischen flussabwärts benachbarten Stationen (**Abb. 78**)?
- Wie verteilen sich **STRAHLER**-, **SHREVE**-, **HORTON**- oder die **KLASSISCHE FLUSSORDNUNG** im Einzugsgebiet eines bestimmten Abschnittes?
- Ist das Einzugsgebiet streng hierarchisch oder befinden sich **Kreisstrukturen** im Gewässernetz?
- Welche Abschnitte gehören zum **Hauptfluss**? Definiert man in einer Flussgebietseinheit nur eine einzige Senke (Mündung des Hauptgewässers oder Passieren der Gebietsgrenze durch Gewässer), ist unmittelbar eine **Fließanalyse** möglich, die durch Fließpfeile (vgl. **Kap. 4.1.3.6**) je Gewässerabschnitt visualisiert werden kann (da bei der Hierarchisierung die Ausrichtung der Gewässerabschnitte zum Gebietsauslass hin keine zwingende Voraussetzung darstellt).

Durch die Verschneidung des Gewässernetzes mit **Wasserscheiden** sind diese mit der Hierarchisierung fixierbar (vgl. **Kap. 4.2.7.3**). Auch hier kann eine visuelle Kontrolle in Form von Fließpfeilen und Überlagerung der Wasserscheide erfolgen.

Das Gewässernetz kann **zwischen zwei Punkten** verfolgt werden. Dadurch ist es möglich, Fragen wie die folgenden, ebenfalls in **Kap. 1.3** gestellten, zu beantworten:

- Wie lang ist die Strecke von Abschnitt A zu Abschnitt B (**Abb. 72**, **Abb. 75**)?
- Wie viele Abschnitte und welche Arten von Knoten und Messstellen befinden sich zwischen diesen Abschnitten?

- In welcher Zeit könnte ein bestimmter Stoff von A nach B transportiert werden?

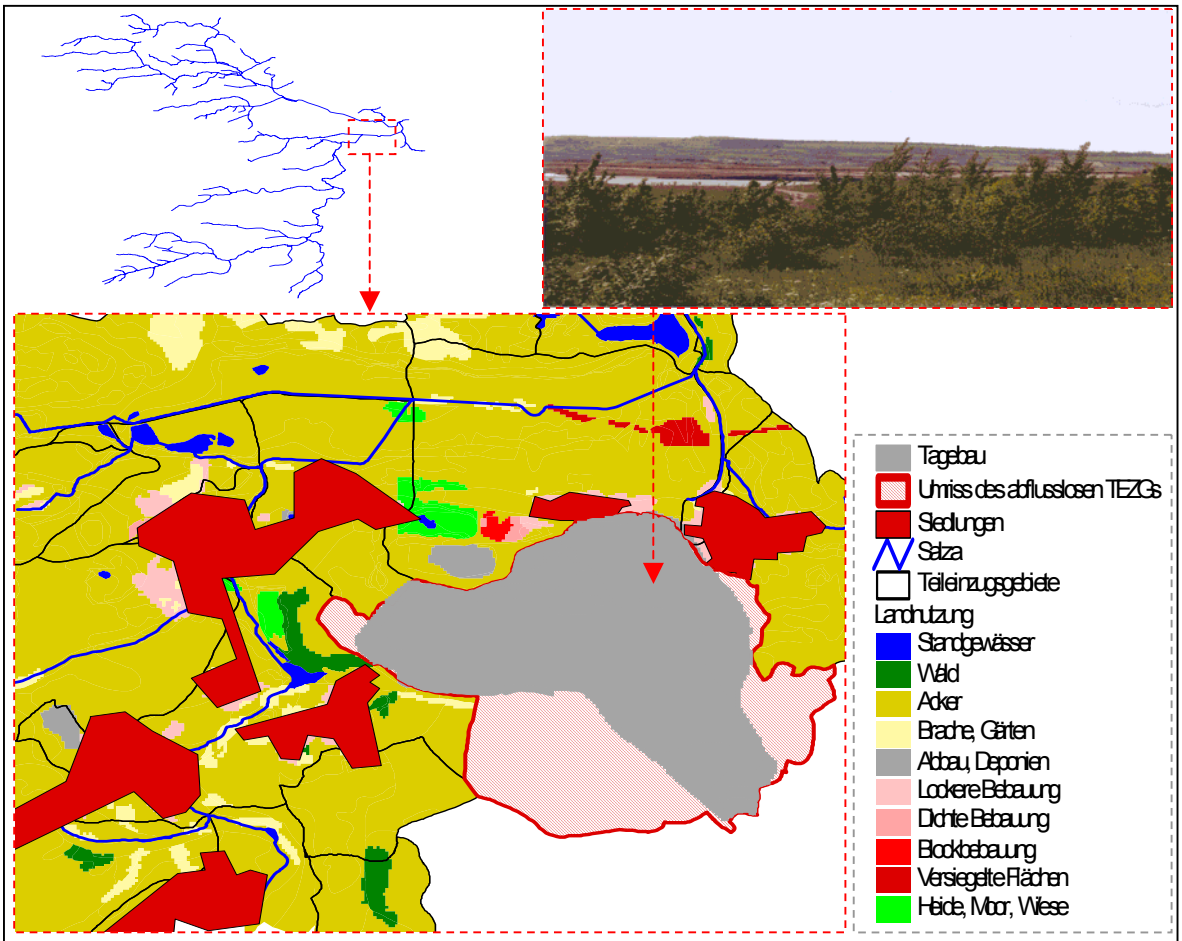


Abb 74: Auffinden oberflächenabfußloser Teileinzugsgebiete (Tagebaurestloch Armsdorf) [Foto: D. WURBS 2002]

Die Verbindung zweier Punkte ist auch Grundlage der Abbildung von **Stationierungsachsen** und einer **Kilometrierung** auf dem Gewässernetz (**Abb. 75**). Die Stationierung von Oberflächengewässern ist ein zentrales Element in der Wasserwirtschaft und eine anerkannte Methode zur Herstellung des Raumbezuges von Objekten, für die keine exakten x- und y-Koordinaten vorliegen, deren Lage bezüglich einer Linie jedoch bekant ist (Angabe der Flusskilometer). Sie eignet sich u. a. für die **Verlinkung mit Gewässerlängs- und Querprofilen**, temporären Probenahmestellen, Gewässergütebereichen oder der Gewässerstruktur. Das System gestattet, Gewässerläufe (**Routen**) interaktiv und sehr effizient zu generieren. Diese Gewässerläufe können anschließend beliebig stationiert (kilometriert) werden.

Der **Ereignismanager** erlaubt, Daten entlang eines Gewässers (z. B. Daten der Erstbeschreibung) unabhängig von der digitalisierten Geometrie zu erfassen, zu recherchieren, zu visualisieren und zu editieren.

Für den Fall einer Abfrage nach den Unterliegern oder der Verbindung zweier Abschnitte lässt sich auch eine **Trendberechnung** durchführen. Die Entwicklung entlang dem Gewässernetz ist durch die implementierte Diagrammdarstellung für mehrere Parameter in einer Abbildung darstellbar (**Abb. 72**, **Abb. 84**).

Es kann auch eine Prüfung des Kriteriums „**Durchgängigkeit** des Flusses“ (Anlage 5 der WRRL) ausgeführt werden. Diese ist im Rahmen der zu erfassenden Belastungen zu bestimmen. Noch ist allerdings durch die WRRL nicht festgelegt, inwieweit diese sowohl flussauf als auch flussab und für welche Organismen bzw. Arten (Durchgängigkeitsklasse) zu bestimmen ist. In Abhängigkeit der Richtung und der zu betrachtenden Durchgängigkeitsklasse kann das Ergebnis ganz unterschiedlich ausfallen. Das Problem lässt sich lösen, indem an den Elementen, welche die Durchgängigkeit potenziell behindern (Schleusen, Wehre, Sohlstürze, Rampen usw.), über ein Attribut (JA/NEIN) für jede Klasse und Richtung die Durchgängigkeit ausgewiesen wird. In dieser Anwendung wurden zunächst 15 Klassen festgelegt, die nach Bedarf jedoch erweiterbar sind. Bei der Durchgängigkeitsprüfung werden dann ein Startpunkt, eine Richtung sowie die Klasse vorgegeben, für die eine Analyse durchgeführt werden soll, und von dort aus alle durchgängigen Abschnitte gekennzeichnet. Das System stoppt an den Stellen, die eine weitere Bewegung verhindern. Änderungen an Bauwerken (z. B. Bau einer Fischtreppe) werden über das JA/NEIN-Attribut vermerkt und dementsprechend bei der nächsten Analyse berücksichtigt.

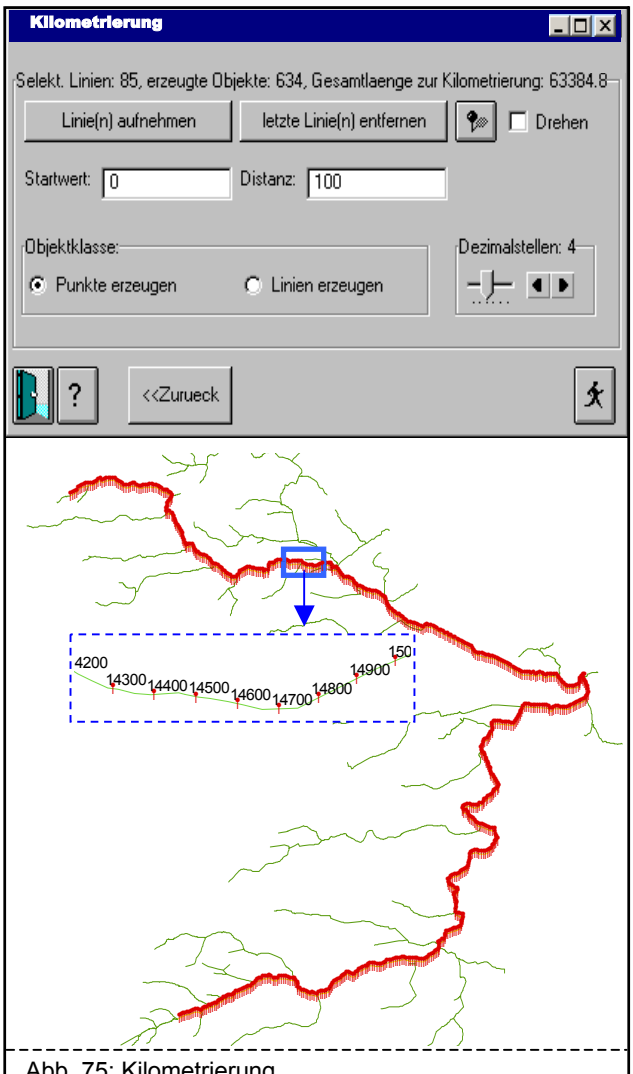


Abb. 75: Kilometrierung
oben: Interface zur Kilometrierung
unten: Kilometrierung im 100m – Abstand auf Basis der Verbindung zweier Punkten (SALZA-EZG: Verbindung zwischen VIEZBACH und obere QUERNE)

Ein hierarchisiertes Gewässernetz kann auch bei der **Abflussberechnung im Gerinne** (engl. „Flood Routing“) nach dem KALININ-MILJUKOV-Verfahren verwendet werden. Aus den morphologischen Parametern des Gerinnebettes sowie den hydrologischen Messwerten wird die Eingangswelle näherungsweise bestimmt und ihre Ausbreitung im Gelände des Abflussbereiches berechnet. Dieses Verfahren dient u.a. zur (Gefahren-) Abschätzung der Verformung einer Welle (z. B. infolge Versagens einer Stauanlage). Dabei wird das Gewässer zunächst in einzelne Teilabschnitte gegliedert, die sich im Allgemeinen von einer Gewässereinmündung zur nächsten erstrecken. Jedes dieser Segmente wird als lineare Speicherkaskade mit Einzellinearspeichern aufgefasst. Dabei gehen sowohl die Translation der Welle als auch die Retention im Flussabschnitt ein. Prinzipiell kann diese Speicherkette dann vom Beginn des Teilabschnittes bis zum unteren Auslauf sukzessive für die einzelnen Speicher berechnet werden [ROSEMANN & VERDAL 1970].

Mit einem unterlagernden DGM lassen sich potenziell von einem bestimmten **Hochwasserstand** betroffene Gebiete oberhalb eines bestimmten Punktes mit einem vom Bearbeiter entwickelten Verfahren ermitteln [SCHORGHOFER & DANIEL 1998]. Dabei werden die

exponierten Abschnitte flussab und in einer vorgegebenen Schrittweite durchlaufen. In Höhe des prognostizierten Wasserstandes erzeugt das Programm eine Peillinie vom Stromstrich aus nach links und rechts in das Gelände hinein (wieder in der vorgegebenen Genauigkeit). Die Koordinaten des Aufeinandertreffens dieser Linien mit der Oberfläche des DGM werden festgehalten. Nachdem alle Oberlieger auf diese Art prozessiert sind, erhält man eine gemittelte Überschwemmungsfläche, indem die gefundenen Schnittpunkte, dem Uhrzeigersinn folgend, in eine Liste für das zu erzeugende Überschwemmungsflächenpolygon gelesen werden (**Abb. 76**).

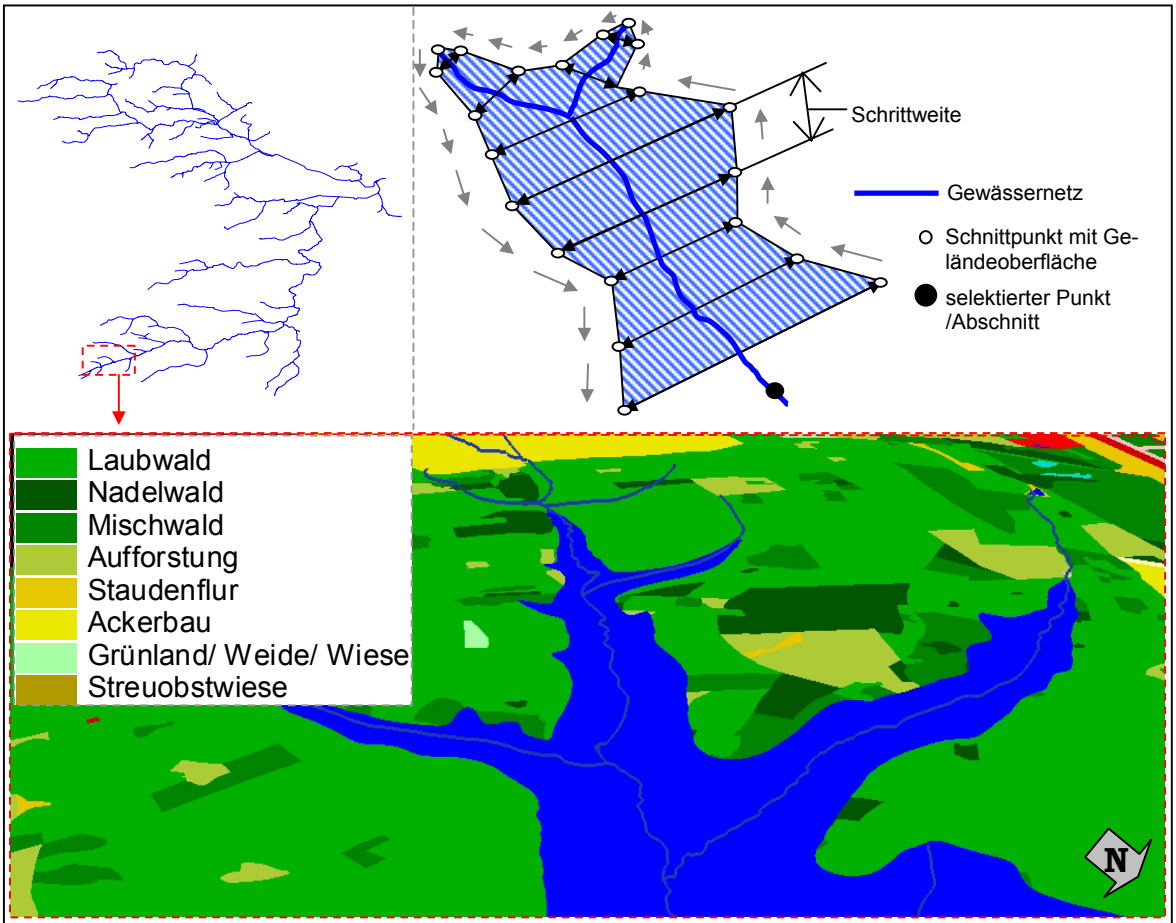


Abb. 76: Ermittlung von Überschwemmungsflächen
oben links: Übersicht, oben rechts: schematische Darstellung des Algorithmus (Beschreibung im Text)
unten: potenzielle Überschwemmungsfläche (blau) mit Landnutzung und Gewässernetz (obere QUERNE)

Ein anderes Modul dient der **Generierung von Gesamt-Einzugsgebieten** an beliebigen Punkten auf Basis der digitalen Teileinzugsgebiete. Nach grafischer Auswahl eines Gewässerabschnittes mit dem Maus-Zeiger, werden alle Teileinzugsgebiete flussaufwärts sowie das Teileinzugsgebiet des ausgewählten Gewässerabschnittes selbst zu einem gesamten Einzugsgebiet aggregiert (**Abb. 68**, **Abb. 77**).

Dabei kann die Beantwortung der folgenden Fragen einen erheblichen Erkenntnisgewinn hinsichtlich der **Eingrenzung und Bewertung potenzieller Maßnahmen** bedeuten:

- Wie viel Wasser kommt an den Gebietsauslässen an?
- Wo sind die potenziell größten Stauvolumen pro Fläche erreichbar?
- In welchen Teilstrecken konzentriert sich das abfließende Wasser?
- Wo führt ein Rückstau zu einer Gefährdung von Siedlungs- oder Verkehrsflächen?

Die Applikation nutzt die zugrunde liegende Hierarchie auch bei der **Interpolation** vorhandener Werte (z. B. Höhen), aus einem Punktthema **auf das Gewässernetz**. Nachdem

diese Punkte den ihnen am nächsten liegenden Vertices im Gewässer zugewiesen sind, werden ihre Werte sukzessive von den Quellen ausgehend zum Auslass hin interpoliert, bis jeder Abschnitt einen Wert für seinen Anfang und sein Ende besitzt. Natürlich steigt die Genauigkeit des Verfahrens mit der Anzahl und Qualität der Punkte und ihrer Attribute. Sinnvoll ist dieses Vorgehen z. B. bei Höhenmodellen geringer Ausgangsdichte oder dem nachträglichen Zuweisen von Höhen und davon abgeleiteter Informationen, wie der Ermittlung des Gefälles einzelner Abschnitte oder der Interpolation desselben auf das gesamte Gewässernetz.

Wurde an einem bestimmten Abschnitt ein **erhöhter Messwert** festgestellt, können die Verteilung des Inputs an den oberhalb liegenden Messstellen mit diesem System sehr

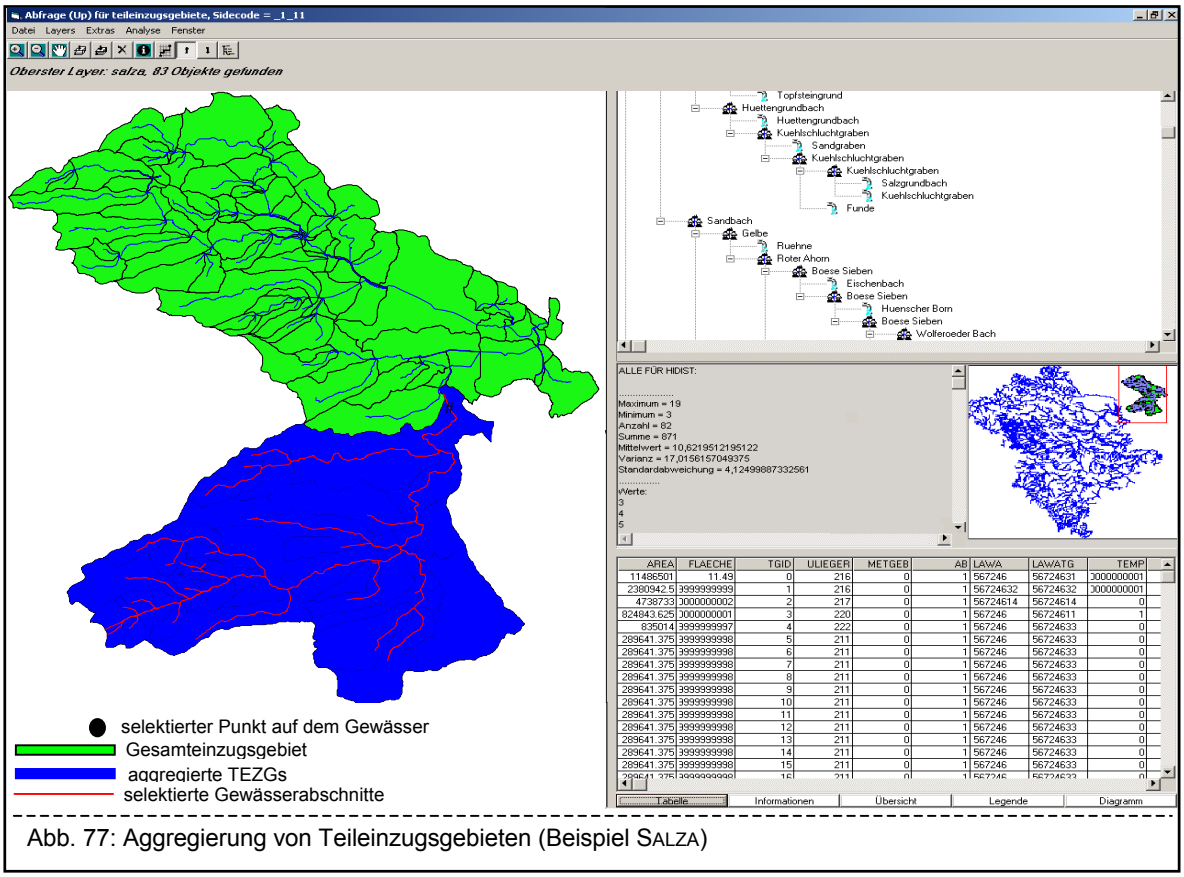


Abb. 77: Aggregierung von Teileinzugsgebieten (Beispiel SALZA)

schnell analysiert, visualisiert und z. B. folgende Fragen beantwortet werden:

- Welches sind die **Hauptquellen**?
- Welche Werte dieses Teils des Messnetzes weichen vom jeweiligen **Durchschnitt** oder einem festgelegten **Zielwert** ab?
- Wo sind die **Haupteintrittspunkte** eines bestimmten Stoffes in das System?

Eine weitere Nutzung der Netzwerkfunktionalität betrifft die Aufstellung von **Bilanzen** (**Abb. 78** bis **Abb. 80**). Da ein beliebiger Knoten alle flussaufwärts liegenden Knoten über das logische Netzwerk kennt, ist es sofort möglich, alle oberhalb liegenden Einleitungen, Entnahmen und Gebietsleistungen zu addieren und auszuweisen.

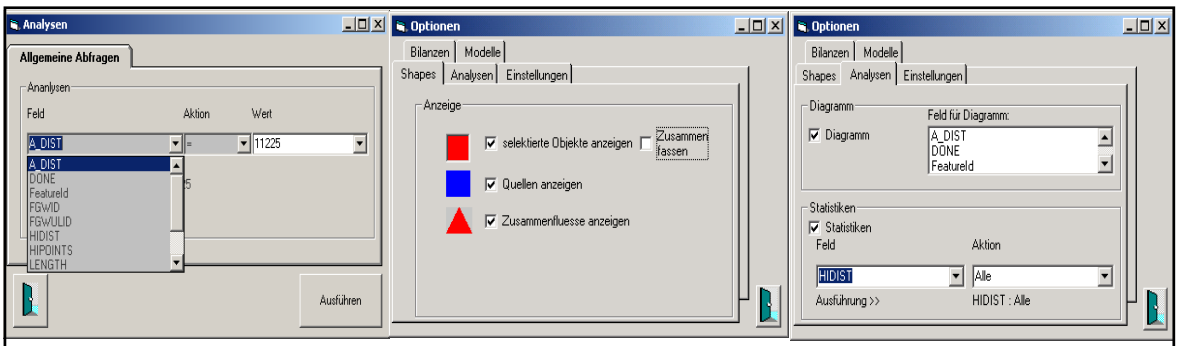


Abb. 78: Auswahl zu den Anzeige-, Bilanzierungs- und Analysemöglichkeiten

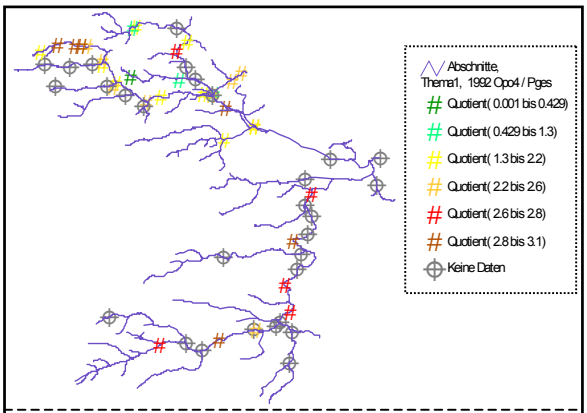


Abb. 79: Ergebnisse der Berechnung des Verhältnisses von Orthophosphat zu Gesamtphosphor (SALZA, Quelle: STAU 1998)

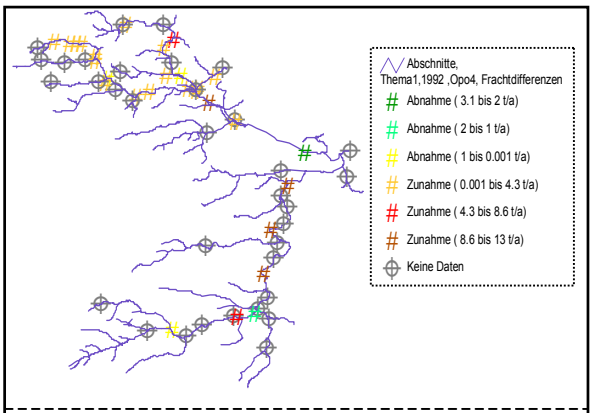


Abb. 80: Konzentrationsdifferenzen zwischen aufeinanderfolgenden Messstationen (SALZA, Quelle: STAU 1998)

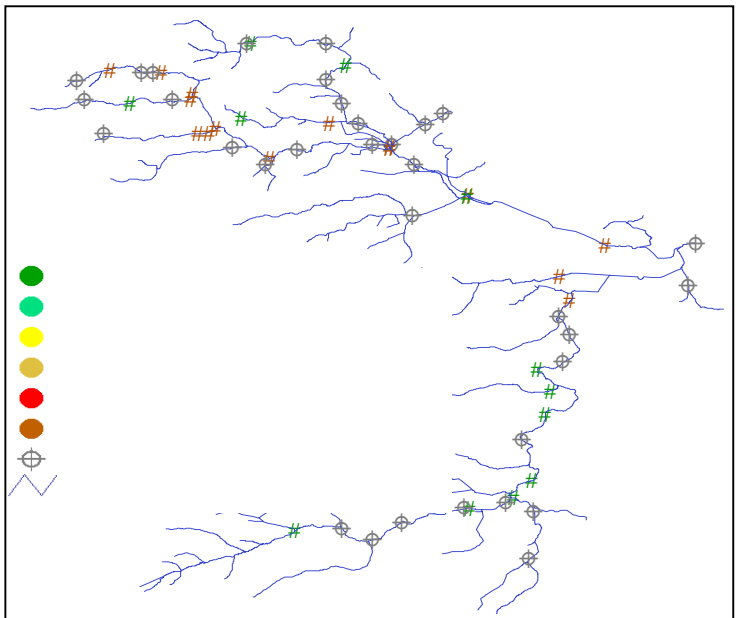


Abb. 81: Zielerfüllung UQS (Umweltqualitätsstandard) 1996 (SALZA)

Die Verbindung der Netzwerkfunktionalität mit Funktionen, die bei Änderung eines Bilanzwertes oder einer Zuordnung bzw. beim Hinzufügen und Löschen von Einleitungen angestoßen werden, ermöglicht die **automatische Neuberechnung der Bilanzwerte**. Ändert man an einem Gewässerabschnitt einen Bilanzwert, werden alle davon abhängigen Bilanzwerte sofort aktualisiert. Die daraus resultierenden Vorteile liegen auf der Hand. Ebenso werden **statistische Werte** wie Summe, Standardabweichung, Durchschnitt, Minimum, Maximum usw. für Abschnitte, Messstellen oder Flächen ober- oder unterhalb eines Punktes oder zwischen zwei Punkten berechnet. Die hierarchiebasierte Einschränkung dieser Analysen unterstützt die systematische Untersuchung und Beobachtung ausgewählter Teilbereiche des Gewässernetzes. Die zugrunde liegende Hierarchie kombiniert Standardabfragen zum Zustand des Gewässers und seines Ein-

zugsgebietes mit der Logik eines baumstrukturierten Netzwerkes. **Abb. 81** zeigt die Zielerfüllungsgrade für Umweltqualitätsstandards im Einzugsgebiet der SALZA.

Die Verbindung des Gewässernetzes mit anderen Punkt-, Linien- und Flächendaten (Anlieger, Profile, Messnetz, vgl. **Kap. 4.2.7.4**) bzw. deren **entkoppelte Hierachisierung** ermöglicht die Bilanzierung von Vorgängen innerhalb des Einzugsgebietes auch unabhängig vom Gewässernetz (**Abb. 73**). So sind per Mausklick z. B. Klima- und andere Messstationen oberhalb eines Punktes auf dem Gewässer oder im Einzugsgebiet abzufragen.

Um die beschriebenen Vorteile dieses Monitoringsystems für Verantwortliche im Fließgewässermanagement auch unter dem ARCVIEW – Framework zugänglich zu machen, wurde ein **Hierarchieexplorer** erstellt, welcher die Struktur des Gewässers in eine alternative Darstellung überträgt (**Abb. 83**).

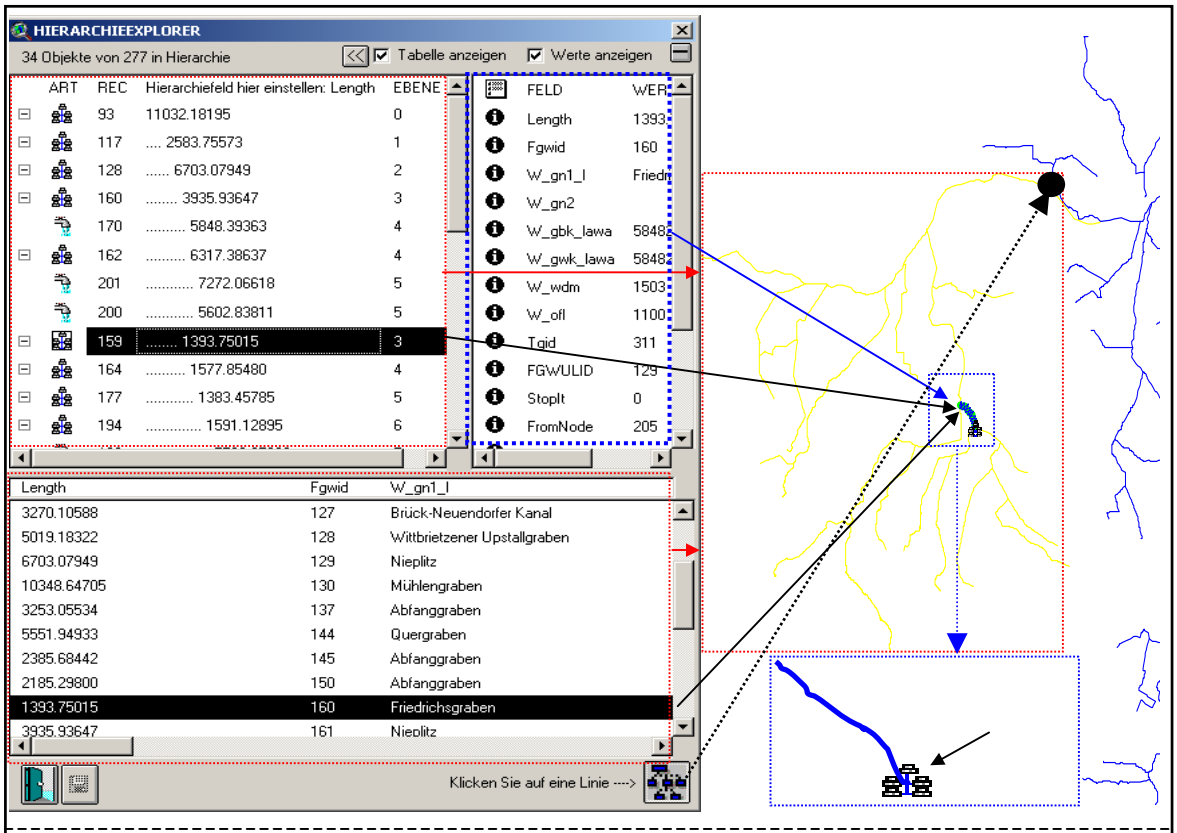


Abb. 82: Hierarchieexplorer in ARCVIEW
linke Bildhälfte:
links oben: Treeview – Hierarchiesicht zum Teilbaum des Gewässernetzes über dem selektierten Knoten
rechts oben: Detailinformationen zum jeweils betrachteten / selektierten räumlichen Objekt (in diesem Fall Gewässerabschnitt)
unten: Tabellensicht
rechte Bildhälfte:
Teilbaum des Gewässernetzes über einem selektierten Knoten (gelb selektiert, Doppelklick auf den Wurzel-Abschnitt)
blau: in Tabelle oder Treeview selektierter Abschnitt, das Icon zeigt die Art des Startpunktes dieser Linie an

In direkter Gegenüberstellung der Karte mit implementierter Tabellensicht, Diagrammen, Statistiken und Detailinformation zu einzelnen Objekten in Kombination mit graphischer Visualisierung verschiedener Informationen ergeben sich auch für den Bearbeiter unter ARCVIEW 3.x verbesserte Möglichkeiten, dem Gewässer innewohnende Informationen zu erfassen, weiterzuverarbeiten oder ggf. zu modifizieren.

Die Kenntnis der Geometrie der Fließgewässer ermöglicht die Erarbeitung von **Quer- und Längsprofile**n der Flussbetten unter Einbezug von Brücken und Verbauungen. Die Darstellung und der Ausdruck dieser Profile im Maßstab sind mithilfe von Spezialprogrammen möglich. Eine solche hierarchiebasierte Applikation wurde erstellt (**Abb. 83**). Dieses Werkzeug kann im Prozess der Ermittlung von Überschwemmungsgebietsgrenzen, zur Verifizierung von Vermessungsdaten und zur Parametrisierung eines hydraulischen Modells (im Präprocessing), sowie zur Überprüfung und Visualisierung hydraulischer Modellergebnisse (Postprocessing) eingesetzt werden. Dabei ist es möglich, Profile, Gewässerlinien und Überschwemmungsflächen zu importieren, Profilgeometrien, Bemessungsabflüsse, Einzelverluste und hydraulische Kennungen zu editieren und die Ergebnis-Profile zu exportieren.

Die Anwendung ermöglicht auch eine **automatisierte Qualitätskontrolle vergebener Codierungen** (z. B. WEG-Schlüssel) mithilfe Clusterbezogener (vgl. **Kap. 4.1.3.8**), von allen Quellen ausgehender aufwärtsgerichteter Abfragen. Die dabei nicht erfassten Abschnitte weisen fehlerhafte Schlüssel auf.

Über die raum-zeitliche Entwicklung einzelner oder mehrere Parameter gibt der **Zeitreihenmanager** mit graphikbasierten Diagrammen Auskunft (**Abb. 84**). Er dient auch der Erstellung und Speicherung von Zeitreihen.

Die Verwaltung des Gewässernetzes erlaubt es auch, **Teileinzugsgebiete** zu einem topographischen Einzugsgebiet auf Basis eines bestimmbaren Attributes zu **aggre-**

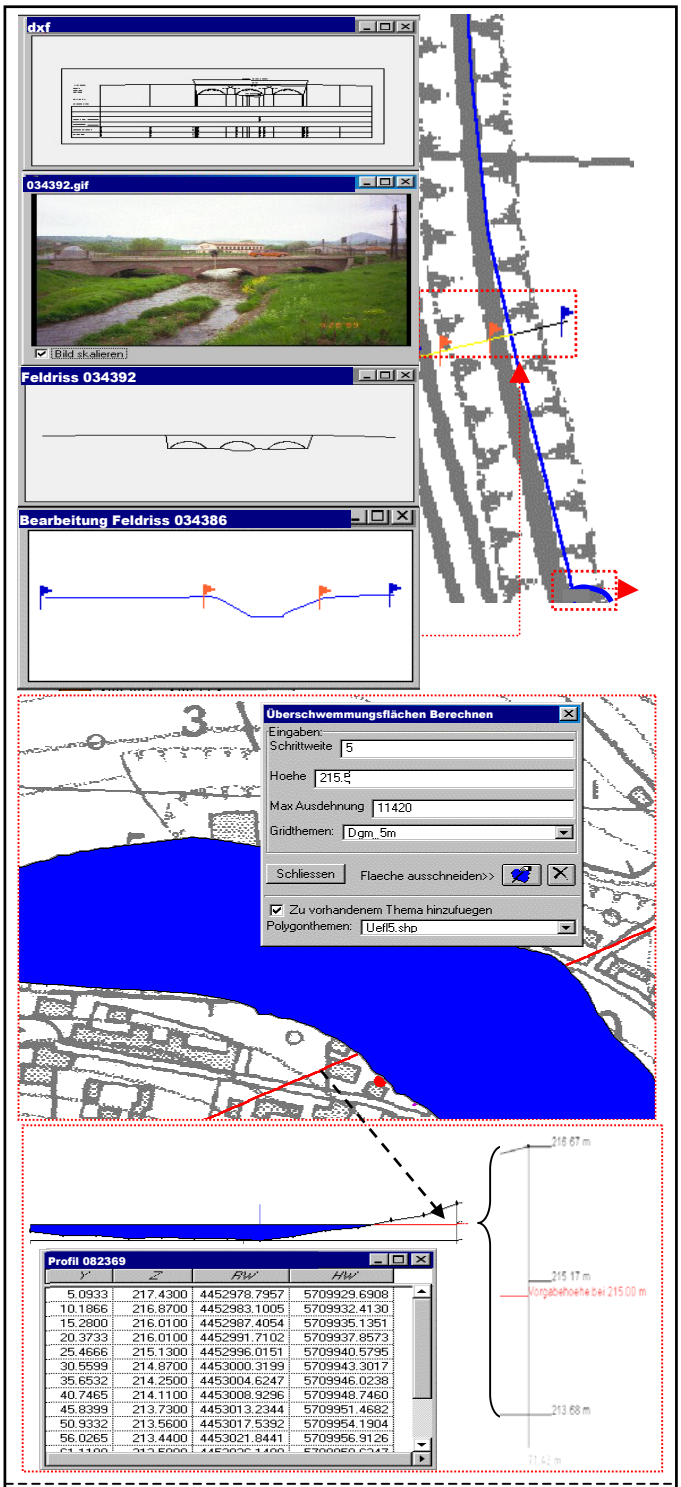


Abb. 83: Hierarchiegestützte Erfassung und Darstellung von:
1. Gewässer-Querprofilen (oben)
2. Überschwemmungsflächen (Mitte und unten)
Die Anwendung ermöglicht die hierarchiegestützte Auswahl von Gewässerpunkten und die automatisierte Berechnung von Querprofilen an diesen Stellen. Informationen zu naheliegenden Bereichen (DXF-Zeichnungen, Fotos) können aufgerufen werden.
Die Überschwemmungsflächen werden nach dem Schema in **Abb. 76** ermittelt und können sowohl in der Aufsicht (Mitte) als auch im Profil dargestellt und exportiert werden.

gieren (der Vorgang der Attributzuweisung wurde unter **Kap. 4.2.7.4** beschrieben). **Oberflächenanalysen von Einzugsgebieten oder von Gemeindegebieten**, oder von einem beliebigen, am Bildschirm eingegebenen Polygon, sind ebenfalls möglich.

Auch der Prototyp eines Kommunalexplorers wurde auf dieser Grundlage erstellt. Er soll die **Navigation innerhalb** der Hierarchie **von Kommunalstrukturen** ermöglichen. Die Zugehörigen Informationen wie Bevölkerungszahlen, Einwohnerdichte, Anschlussgrad, Trinkwasserbedarf usw. werden, sofern sie vorhanden sind, dynamisch abgerufen und stehen weiteren Auswertungen zur Verfügung.

Natürlich besteht auch die Möglichkeit zur effektvollen **Integration verschiedener Medien** in die hierarchiebasierten Datensichten. So z. B. bei der Verbindung zweier Abschnitte und aller sie unterlagernden Datenschichten. Ein einfaches Beispiel ist die filmische Abfolge von Bildern und Informationen zu den passierten Gewässerabschnitten und deren Umgebung, wie sie beispielsweise im Tourismus gut einzusetzen wäre.

Durch die Verbindung der einzelnen Informationsträger wird auch hier die Arbeit mit den Gewässerdaten intuitiver und ergonomischer. Die Kopplung mit anderen geographischen Informationsschichten wie Einzugsgebieten oder dem Messnetz erhöht den Optimierungsgrad von Gewässermanagement bzw. –monitoring.

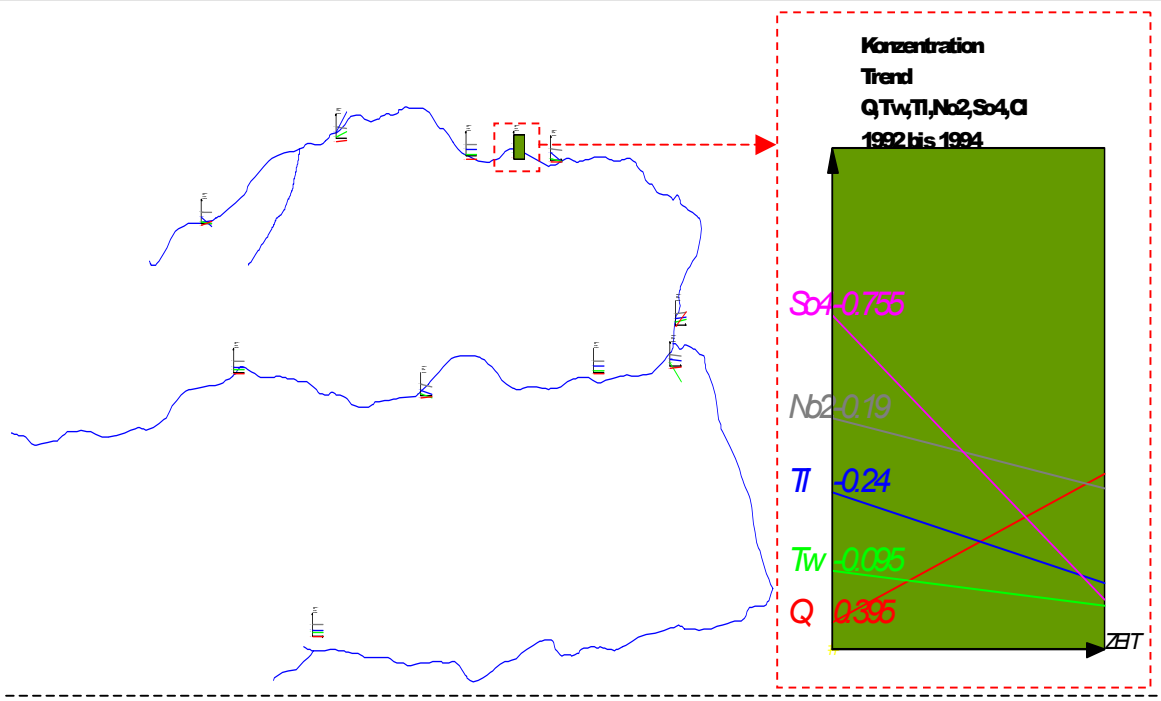


Abb. 84: Trenddiagramme für verschiede Parameter (TEG BÖSE SIEBEN / SALZA, QUELLE: STAU 1998)
Die Diagramme zeigen die zeitliche Entwicklung verschiedener Parameter an den Messstellen als lineare Trends

4.3.2 Anwendung in einem datenbankgestützten, hierarchiebasierten, internetfähigen Fließgewässer-Informationssystem

Nach Artikel 14 derEU-WRRL [WRRL 2000] ist der Öffentlichkeit umfassend Zugang zu Hintergrundinformationen über Grundwasser und Oberflächengewässer in geeigneter Form durch Medien mit hohem Verbreitungsgrad – insbesondere das Internet – zu ermöglichen [BUND 2002]. Die LAWA hat deshalb 2001 den Ausschuss „Daten“ ins Leben gerufen, dessen Hauptaugenmerk auf der Internet-Technologie liegt, der technischen Innovation der letzten Jahrzehnte, welche die Welt wohl am nachhaltigsten beeinflusst hat [ECO 1993]. Zur Übertragung der Konzeptionen aus der Anwendung von **Kap. 4.3.1** auf das Internet, ist eine Implementierung von Interaktionsmöglichkeiten des Benutzers mit den dargestellten Webseiten notwendig. Ein Weg besteht in der Erzeugung dynamischer Webinhalte. Die meisten professionellen Anwendungen mit Datenbankanbindung, Gästebüchern, Suchmaschinen oder Onlineshops kommen heute nicht mehr ohne dynamisch generierte Webseiten aus. Im Gegensatz zu statischen Webseiten, die stets in unveränderter Form auf einem Webserver abrufbereit stehen, werden sie erst auf eine Benutzeranforderung (Interaktion) hin erzeugt. Die Erzeugung dynamischer Webseiten ist vor allem dann notwendig, wenn nicht von vornherein feststeht, welche Informationen durch eine Anfrage bereitgestellt werden. Das trifft naturgemäß vor allem auch auf die Nutzung einer Datenbank und von Geodaten über das Internet zu, wo die Suchergebnisse erst dann generiert werden können, wenn der Nutzer eine Anfrage – z. B. Zoomen oder Verschieben des Kartenausschnittes - gestellt hat.

Eine der Prämissen der Arbeit ist die Anwendbarkeit der Ansätze auf einen möglichst weit gefächerten Bereich technischer Voraussetzungen und inhaltlicher Fragestellungen. Dies bedeutet, eine allgemein verfügbare und hinreichend akzeptierte Software-Konstellation zugrunde zu legen. Die vorliegende internetorientierte Applikation sollte dem 3-Schicht-Modell (Datenbankschicht - MySQL, Applikationsschicht - PHP, UMN-MAPSERVER [UMN 2001] und Präsentationsschicht – Browser) folgen und benötigt darüber hinaus einen Webserver (auch Internet- oder HTTP-Server, in diesem Fall APACHE [APACHE 2003]). Die Verteilung der Komponenten und die Arbeitsweise des Systems sind in **Abb. 85** schematisiert. Mithilfe eines Webservers kann der Browser mit dem Server Kontakt aufnehmen. Er bearbeitet Anfragen, erstellt neue HTML-Seiten und sendet diese wieder an den Nutzer (Client) zurück. Somit ist gewährleistet, dass die Applikation auf jedem handelsüblichen Browser zu betrachten ist. Dabei können dem Webserver mit dem URI (Uniform Resource Identifier) Parameter übergeben werden, die kartographischen Applikationen Interaktionen des Nutzers, wie etwa Änderung des Bildausschnittes, Maßstabsänderungen oder Datenbankabfragen, an den Server übermitteln. Auch in dieser Anwendung (**Abb. 86** bis **Abb. 90**) sollte eine Verbindung verschiedener Sichten auf die Datenbasis zu einem beliebigen Gewässernetz erreicht werden. Alle Bestandteile sind Opensource-Produkte [OGC 2001] und erfahren eine seit Jahren sehr hohe und rasant anwachsende Verbreitung.

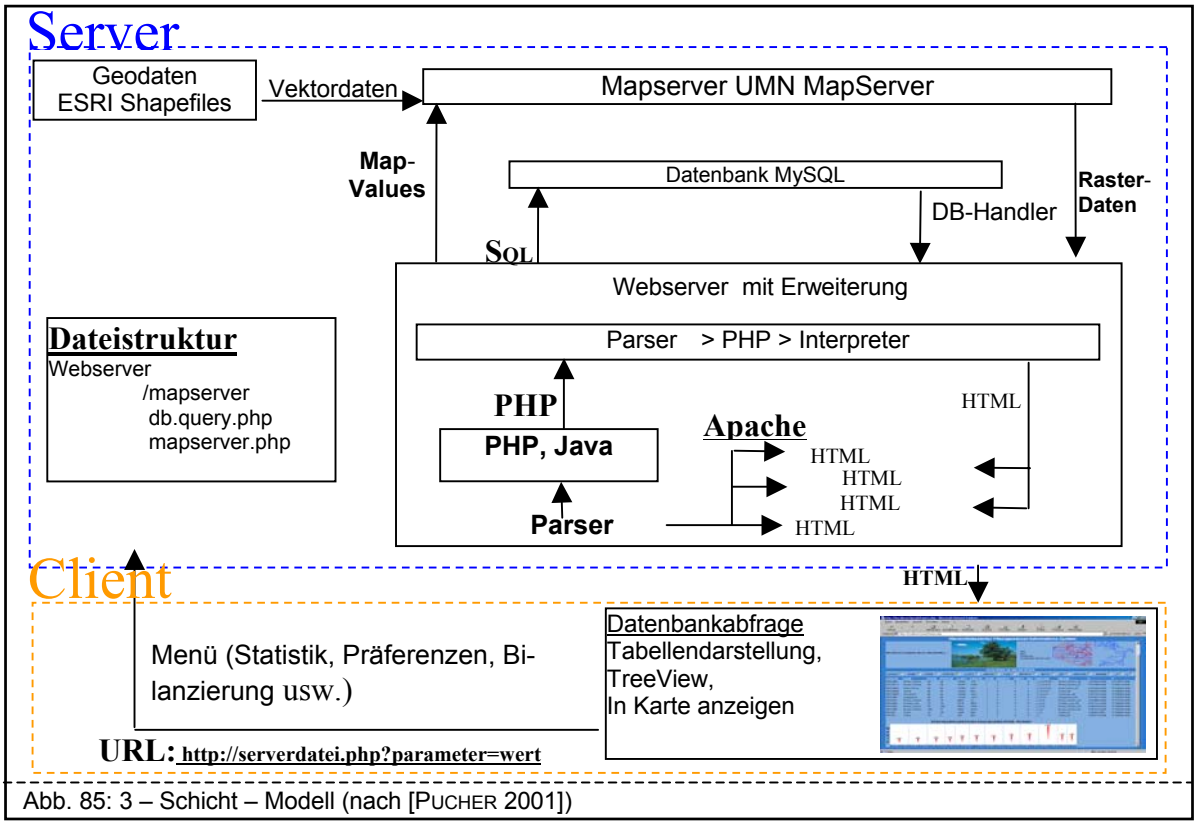


Abb. 85: 3 – Schicht – Modell (nach [PUCHER 2001])

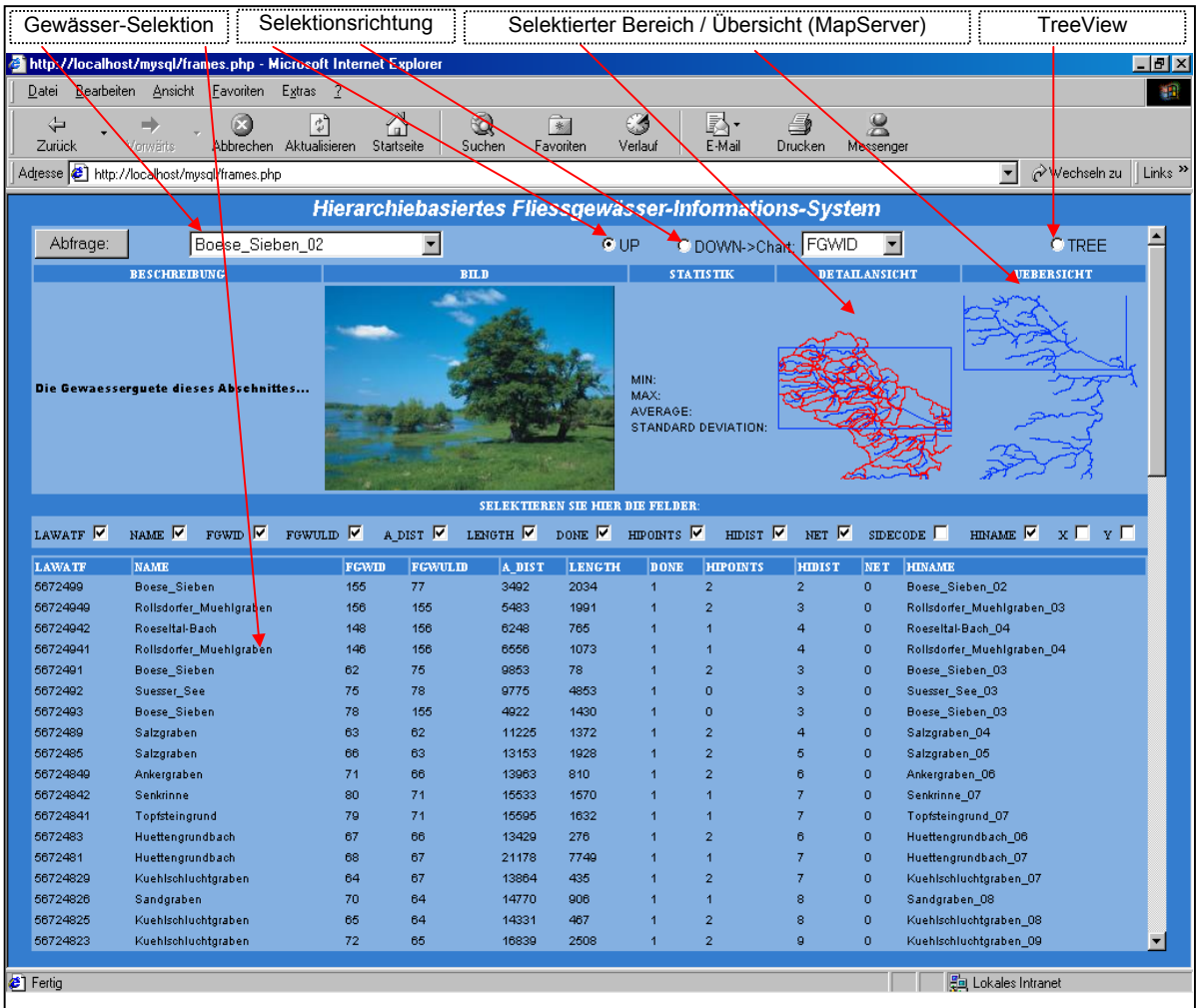


Abb. 86: Internetapplikation mit abwärts gerichteter Abfrage

MySQL [MYSQL 2002] ist ein kompaktes und leistungsfähiges relationales Datenbanksystem, das mit seiner Performance, Kostenfreiheit für nichtkommerzielle Anwendungen und Portabilität auf unterschiedliche Rechnerplattformen große Popularität erreicht hat. Auf Grund seiner Flexibilität ist MySQL bei der Entwicklung von Datenbankanwendungen oft eine ernsthafte Alternative zu großen kommerziellen Datenbanken wie ORACLE, DB2 oder INFORMIX. Mit der Version 4.1 wurden auch räumliche Indices in Verbindung mit dem neuen Datentyp Geometry eingeführt [MYSQL 2002].

PHP-Scripte werden auf dem Server ausgeführt – im Gegensatz z. B. zu clientseitigen JavaScripts und JAVA-Applets. Der HTML-Code wird beim Abruf der Webseite, wie bei normalen Seiten auch, 1:1 an den Client geschickt. Die Syntax ähnelt C, JAVA oder Perl. PHP wird durch eigene Funktionen erweitert, wie z. B. Kommandos zur Integration von Datenbanken.

Für die geometrische Komponente wurde der MAPSERVER der UNIVERSITY OF MINNESOTA (UMN[40]) gewählt, der in Kooperation mit der NASA und dem MINNESOTA DEPARTMENT OF NATURAL RESOURCES (MNDNR) entwickelt wurde. Im Laufe der Zeit erfolgten Erweiterungen durch das MNDNR und das MINNESOTA LAND MANAGEMENT INFORMATION CENTER (LMIC) sowie verschiedene Firmen, Institute und Organisationen. Mithilfe von MAPSCRIPT [DM SOLUTIONS 2001] kann durch diverse Script-Sprachen wie z. B. PERL, PYTHON, TK/TCL, PHP oder JAVA auf die Funktionalitäten der MAPSERVER-C-API zugegriffen werden, unter denen die Folgenden genannt seien:

- Verarbeitung von Vektor- und Rasterdaten
- maßstabsabhängige Visualisierungen
- automatisierte Beschriftung räumlicher Objekte
- automatische Erstellung von Zeichenklärung und Maßstabsleiste
- Selektion von Geoobjekten mittels Punkt- oder Flächenselektion
- Unterstützung von True-Type-Vektorschriften
- freie Gestaltungsmöglichkeiten der Ausgabeseiten
- Möglichkeit der Kachelung von Vektor- und Rasterdaten
- Indexierung räumlicher Daten.

Dieser Funktionsumfang kann mithilfe von Eingriffen in den Programmcode oder durch Hinzufügen von Funktionen (z. B. MAPPLET [LIME 2001]) erweitert werden. Über die Verwendung geeigneter Datenbankschnittstellen ist es möglich, Sachdaten direkt aus einem Datenbanksystem zu extrahieren und zu visualisieren. Im 3-Schicht-Modell stellt der UMN MAPSERVER die Applikationsschicht dar und erstellt aus einem Geomodell die kartographische Ausdrucksform. Hierzu werden vektorielle Objekte aufgrund ihrer Attributierung symbolisiert und visualisiert. Das so erstellte Kartenbild wird in ein Rasterbild umgewandelt und in Form einer HTML-Seite an den Client verschickt. Sendet dieser eine Anfrage zur Änderung des Kartenbildes, so wird aus dem Geomodell eine völlig neue Darstellung generiert und übersendet. Clientseitige Lösungen - etwa mittels JAVA-Applets - bieten nach dem Laden der Geodaten eine vergleichsweise bessere Performance als serverseitige Mapserver, da nicht bei jeder Interaktion Verbindung mit dem Server aufgenommen werden muss [FRIEDRICH 2002]. Daher wurde auch ein JAVA-Applet auf Basis der GEOTOOLS [GEOTOOLS 2003] entworfen (**Abb. 87**). In diesem können über die Selektion inner-

[40] Eine umfangreiche Analyse des UMN-MapServers sowie des ArcIMS™ (ESRI) findet sich in [FRIEDRICH 2002].

halb der Karte ebenfalls hierarchische Abfragen durchgeführt werden. Dabei bestätigten sich Untersuchungen zur Geschwindigkeit von server- und clientseitiger Anwendungen [FRIEDRICH 2002].

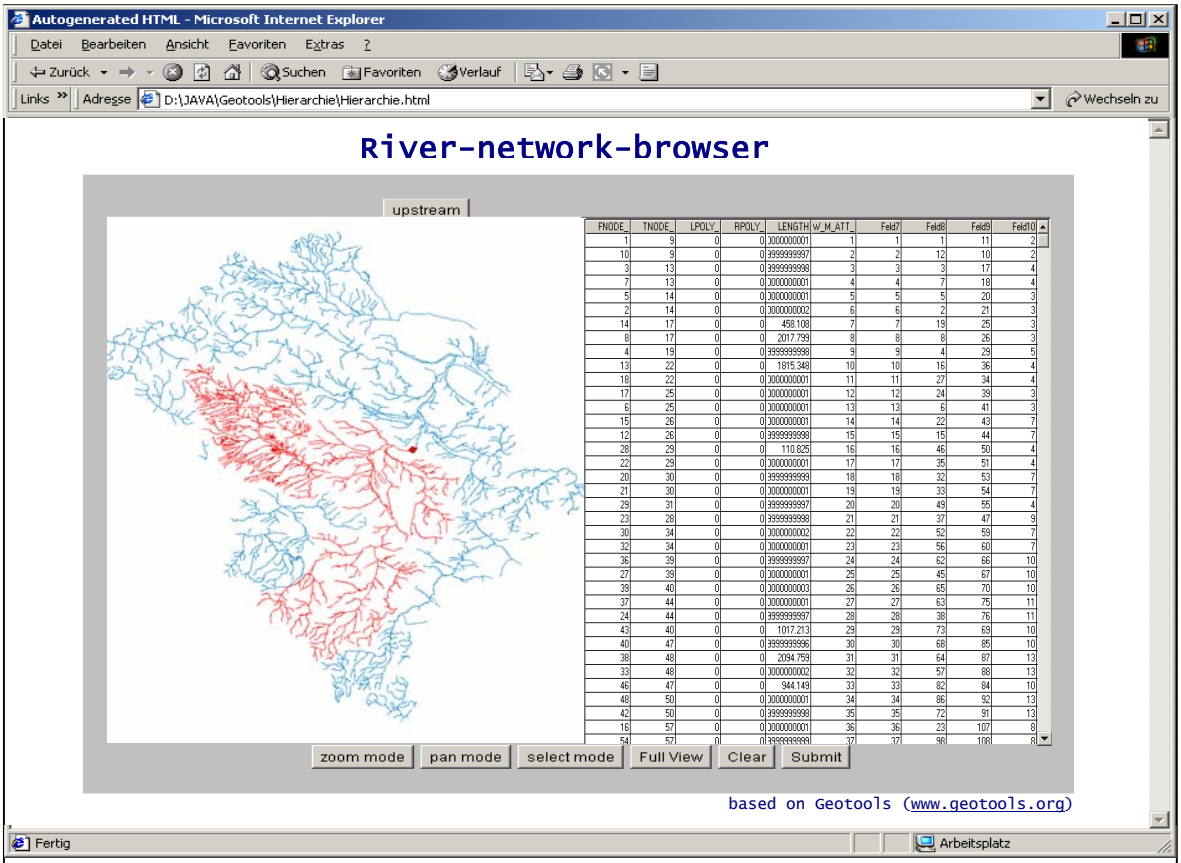


Abb. 87: JAVA-basierter Gewässerbrowser (EZG der UNSTRUT)

Demgegenüber steht der Nachteil, dass zur Darstellung der Abbildung häufig Plug-Ins installiert werden müssen, oder der Browser JAVA-fähig sein muss. Hersteller weitverbreiteter Geoinformationssysteme – exemplarisch seien hier ESRI, MAPINFO und SICAD genannt – bieten Mapserver-Erweiterungen ihrer GIS-Produkte an, die nach serverseitigen Verfahren arbeiten. Daneben existieren diverse frei erhältliche Programme mit Mapserver-Funktionalitäten.

Das System besteht somit aus frei erhältlichen Software-Modulen, die bei Bedarf im Einzelnen ausgetauscht werden können, ohne dass das Gesamtkonzept verändert werden muss. Diese Konfiguration ist einerseits eine ökonomische Variante und andererseits eine zufriedenstellende Lösung in Hinblick auf die Erweiterbarkeit des Projektes.

Umsetzung am Beispiel der SALZA

Hauptfunktion der Internetseiten sollte sein, gewässerbezogene Informationen aus einer Datenbank zu selektieren, zu verarbeiten und anschließend wahlweise in einer Karten-, Diagramm-, Treeview- und/oder Tabellendarstellung zu visualisieren. Der Mapserver sollte dabei den jeweils von der Selektion betroffenen Raumausschnitt abbilden. Für die Tabellendarstellung werden sowohl die stromauf- als auch die stromabwärtige Selektion ermöglicht, die Selektion erfolgt über den Gewässernamen innerhalb einer Auswahlliste bzw. durch Klick in die Tabelle, die Karte oder den Treeview.

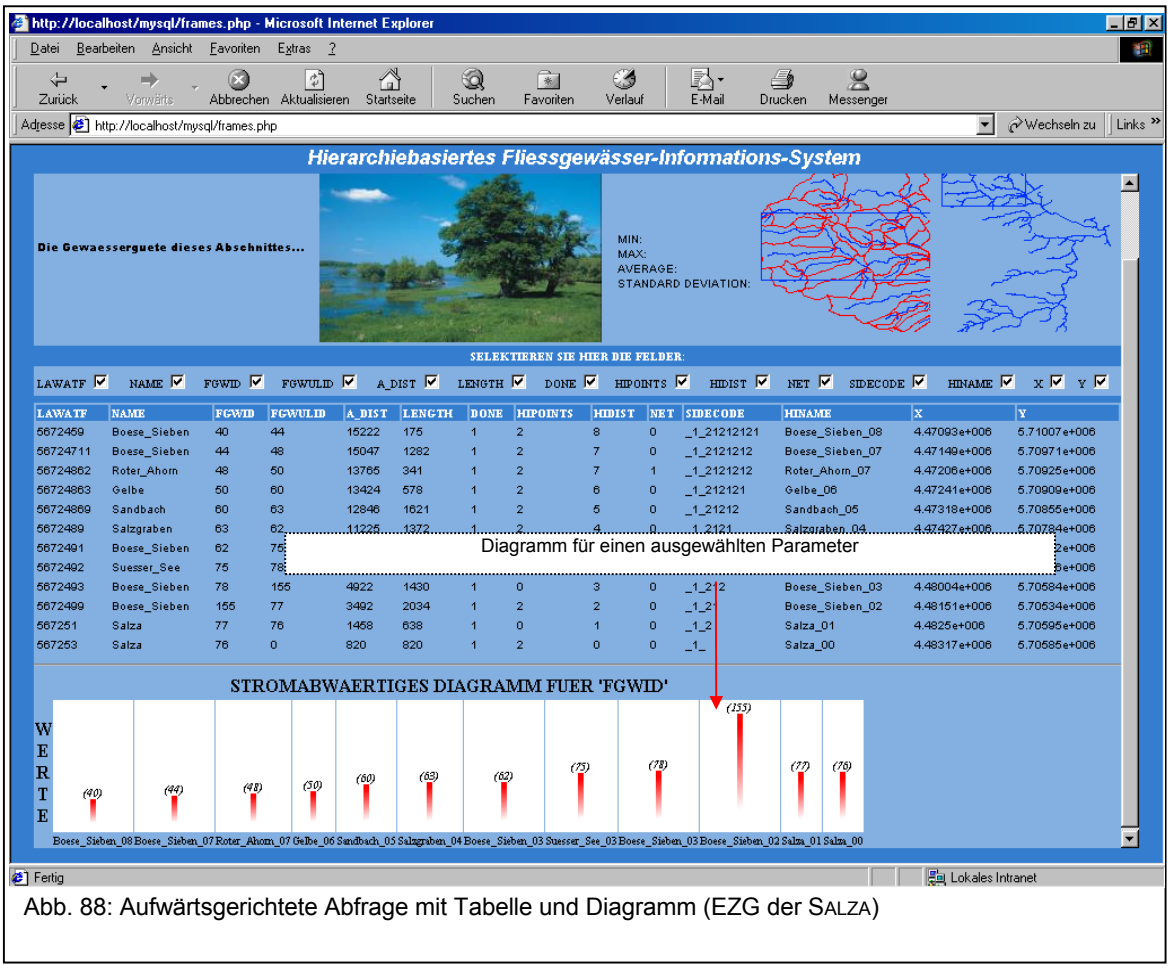


Abb. 88: Aufwärtsgerichtete Abfrage mit Tabelle und Diagramm (EZG der SALZA)

Die Abfragen auf die MySQL-Datenbank werden mittels PHP umgesetzt. Um die Shape-Attribute nach MySQL zu transferieren, wurde eine VISUALBASIC-Applikation erstellt. Dabei gab es im Wesentlichen folgende Anforderungen an die Internet-Anwendung:

Suche in der Datenbank: Als Ergebnis der Suche nach dem Schlüssel und der Übergabe der Selektionsrichtung wird der Inhalt der Datenbank zu dieser Anfrage in der gewünschten Form dargestellt (vgl. **Code 24**).

Abbildung in der Tabelle: In der Tabellensicht werden sowohl die Ergebnisse der auf- als auch der abwärts gerichteten Abfrage dargestellt (vgl. **Code 24** bis **Code 27**, **Abb. 86**, **Abb. 88**).

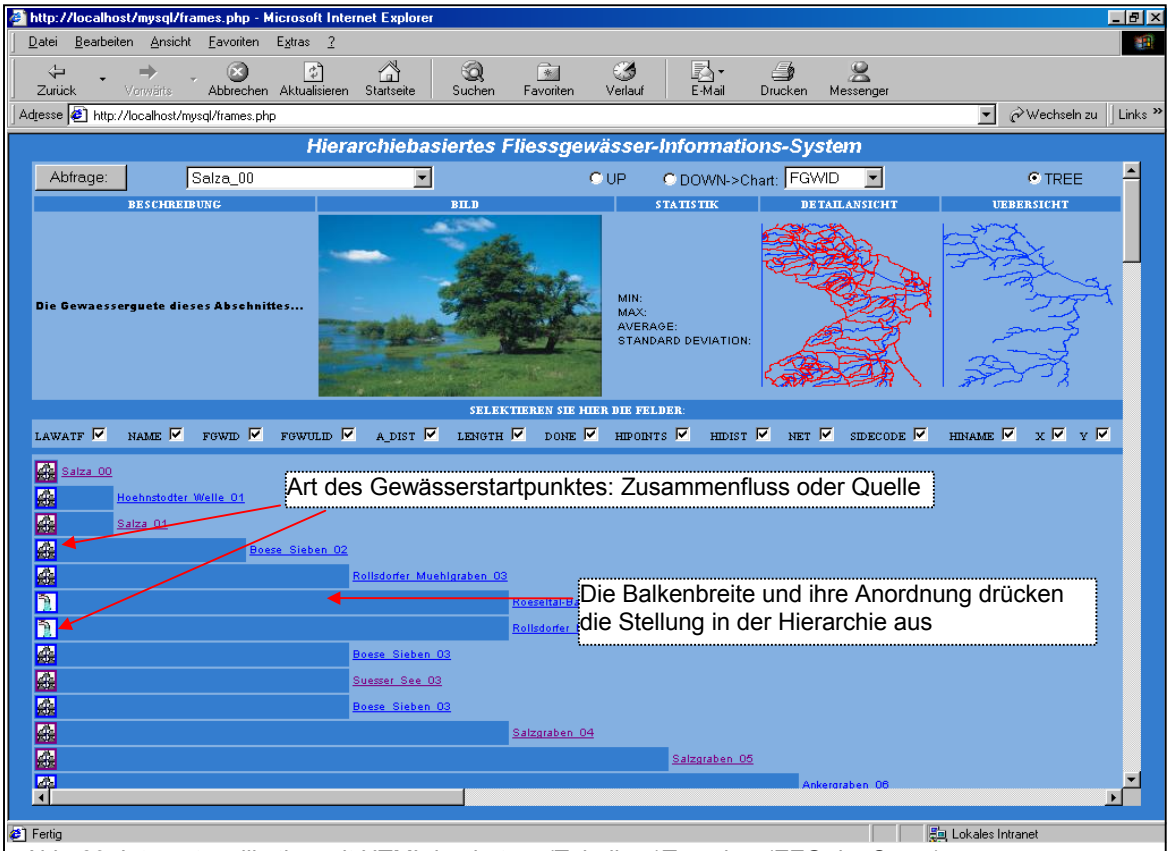


Abb. 89: Internetapplikation mit HTML-basiertem (Tabellen-)Treeview (EZG der SALZA)

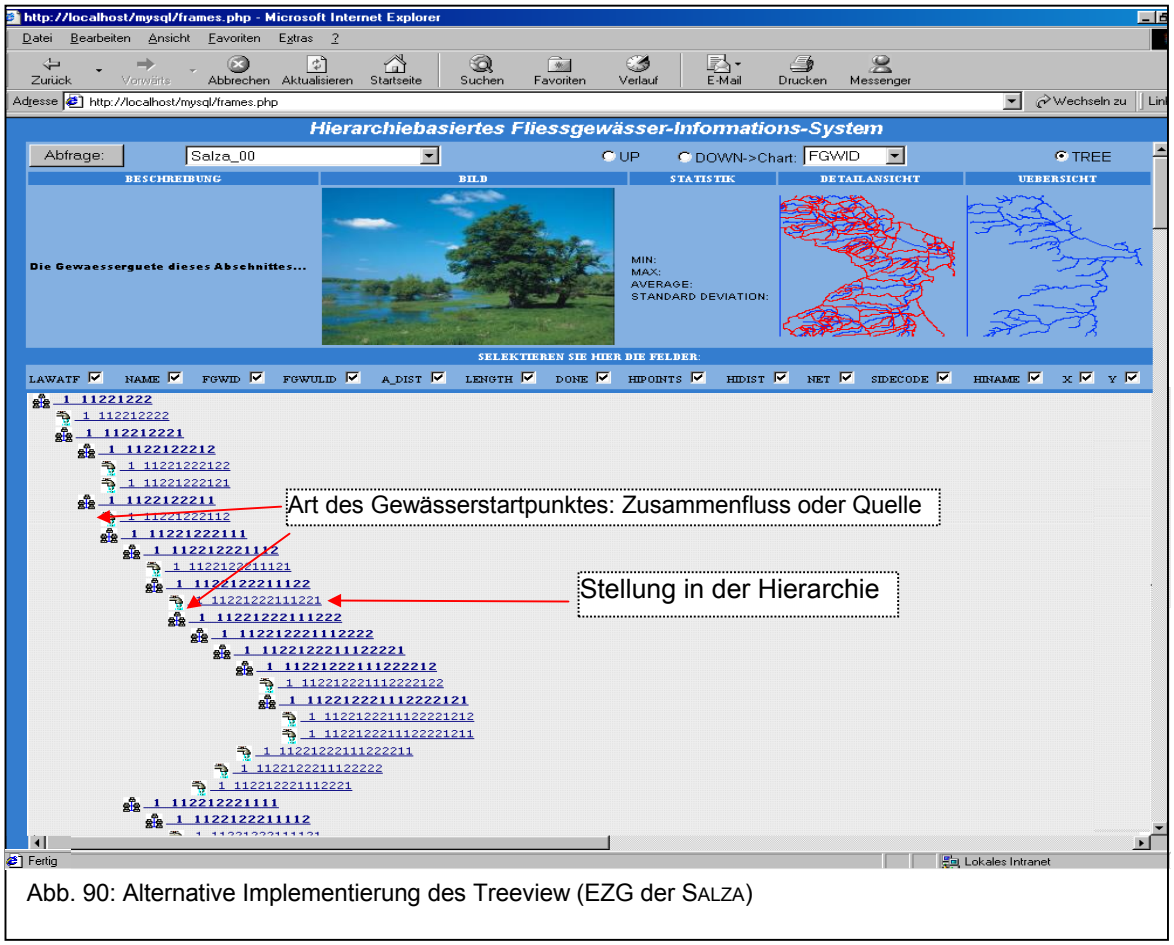


Abb. 90: Alternative Implementierung des Treeview (EZG der SALZA)

Abbildung im Treeview: Das Gewässernetz wird hier in seiner hierarchischen Struktur, als HTML-Tabelle (**Abb. 89**, **Code 28**) oder durch eine rekursive PHP-Funktion (**Abb. 90**, **Code 29**) visualisiert (vgl. **Kap. 4.3.1**).

Berechnung von Statistiken, Bilanzen und morphologischen Parametern: Exemplarisch wurde die Möglichkeit der Berechnung von Statistiken für selektierte Bereiche des Gewässernetzes nach dem Vorbild des in **Kap. 4.3.1** beschriebenen Managementsystems eingebunden.

Darstellung der Entwicklung eines Attributes entlang dem Gewässerverlauf: Neben der Abbildung der Entwicklung eines Parameters entlang dem Gewässer in Tabellenform ermöglicht eine PHP-Funktion zur Erzeugung von Balkendiagrammen auf Grundlage eines Scriptes von P. DAVIS [DAVIS 1998] die Verfolgung der Entwicklung eines Parameters auf dieser Strecke (**Abb. 88**).

Diese Darstellungsformen lassen sich durch weitere Informationen ergänzen wie beispielsweise mit der Visualisierung der Art des Linienanfangs (Quelle, Pseudonode, Zusammenfluss) mithilfe des Attributes „HIPOINTS“ (vgl. **Kap. 4.2.7.1**, **Kap. 4.3.1**) oder der Symbolisierung der Gewässergüteklasse. Die Funktion sollte dabei den Anforderungen folgen. Als Entwurf bietet sich die unter **Kap. 4.3.1** beschriebene Anwendung an.

5. Zusammenfassung und Ausblick

Die Europäische Wasserrahmenrichtlinie schreibt den Einsatz Geographischer Informationssysteme als Basisinstrument für die Datenverwaltung fest, wobei die Reichweite dieser Vorgabe nicht bestimmt ist. In den Verwaltungen werden neben umfangreichen Strukturierungs- und Codierungsaktivitäten die Datenaufbereitung von digitalen Gewässernetzen, oberirdischen Einzugsgebieten (> 10 km), Grundwasserkörpergrenzen usw. vorangetrieben. Darüber hinaus müssen die Daten zur Erstbeschreibung dieser räumlichen Elemente erhoben werden. Mit der entstehenden Datenflut ergeben sich Probleme in der GIS-basierten Datenhaltung. Die in dieser Arbeit vorgestellten Programme zur weitestgehend automatisierten Analyse, Korrektur und Hierarchisierung digitaler Gewässernetze sowie der Verwendung entstehender logischer Netzwerke im datenbankgestützten, internetfähigen Fliessgewässermanagement wurden in Zusammenarbeit mit verschiedenen Institutionen und Firmen an unterschiedlichen Datengrundlagen entwickelt. Sie sollen die hohen Anforderungen hinsichtlich des komplexen Datenmanagements sowie der Entscheidungsvorbereitung für die geforderte flussgebietsbezogene Bewirtschaftungsplanung unterstützen.

Räumlicher Ausgangspunkt dieser Arbeit war das Einzugsgebiet der SALZA mit in spatiotemporal hoher Dichte vorliegenden Daten. Die einzelnen Problemlösungen wurden jedoch immer wieder auch an anderen Testgebieten validiert. Im beständigen Austausch mit den dortigen Bearbeitern und durch die immer wieder neuen Sichtweisen flossen viele, oft auch gegensätzliche Hinweise und Verbesserungsvorschläge zu den verschieden Werkzeugen in den Entwicklungsprozess ein. Die entstandenen Programme sind ein Kompromiss aus diesen Einflüssen und den Vorgaben zur vorliegenden Arbeit, die sich als solche bei ihrem Einsatz vielfach bewährt haben. Effektivität und Qualität der Programmlösungen wurden daher schrittweise verbessert. Die einzelnen Arbeitsschritte wurden immer automatisiert, d.h. mittels geeigneter Algorithmen innerhalb eines GIS-Programmes umgesetzt.

Um die Hierarchisierung von Gewässernetzen möglichst optimal zu implementieren, wurden verschiedene Flusskodierungssysteme untersucht, ggf. implementiert oder entwickelt. Einen Überblick zu verschiedenen Ordnungssystemen und ihrer Eignung hinsichtlich der Arbeit zugrunde liegender Kriterien geben **Kap. 2.1.5** und **Tab. 4.1**. Dabei zeichnen sich der GENERISCHE VERZWEIGUNGSCODE und besonders das SYSTEM VERSCHACHTELTER HIERARCHIEN, die beide im Rahmen dieser Arbeit entwickelt wurden, durch eine sehr gute Eigenschaftskonstellation aus. Beide können in verschiedenen Anwendungen gute Dienste leisten. So ist es z. B. nicht notwendig, bei örtlichen Fragestellungen das gesamte zusammenhängende Flussnetz zu durchlaufen. Alle Abfragen sind auch auf Teildatensätze anwendbar, wodurch sich die Prozesszeiten verkürzen und die Übersichtlichkeit der Ergebnisse erhöht. Die einfach zu erzeugenden SQL-Abfragen in jede Richtung ermöglichen den Einsatz entweder nur im RDBMS/RDB, nur im GIS oder in einer Kopplung dieser Komponenten, unabhängig von exklusiven GIS- oder Datenbankroutinen wie etwa Topologie- oder Netzwerkfunktionen, wie sie beispielsweise im UTILITY NETWORK ANALYST unter ARCGIS™ zu finden sind. Die Codierung ist intuitiv und die SQL-Statements können einfach vom Bearbeiter bzw. durch automatisierte Routinen im GIS oder RDBMS implementiert werden. Mit der Einbindung verschiedener Cluster kann das System VERSCHACHTELTER HIERARCHIEN theoretisch auch global angewendet werden. VERZWEIGUNGSCODE und VERSCHACHTELTE HIERARCHIEN sind in der Übertragung auf alternative Darstellungs-

und Analysevarianten der Gewässerhierarchie sehr leistungsfähig. Sie lassen sich relativ einfach als Treeview abbilden oder zur tabellenbasierten Visualisierung in HTML nutzen und können beliebig erweitert und ergänzt bzw. verfeinert werden. Diese beiden Flusskodierungssysteme enthalten aufgrund ihrer Struktur und Anlage die Möglichkeit der Kopplung mit einer Vielzahl von Informationen. Fragen nach zusammenhängenden Einzugsgebieten, nach flussauf- und flussabwärts gelegenen Verbindungen, nach der Flusslänge, Fließrichtung usw. sind aus der Kombination der Metadaten mit anderen Attributen ableitbar.

In Hinblick auf die Optimierung von Einzugsgebietsmanagement und -monitoring wurden verschiedene Verfahren zur räumlichen und inhaltlichen Erweiterung der hierarchischen Metadaten wie kumulative Werte (**Kap. 4.2.7.2**), Einbindung von Sonderfällen wie Wasserscheiden, Ringstrukturen oder Deltabereiche, die Hierarchisierung flächen- und punkthafter Elemente und ihre Verbindung zum Gewässernetz, Verbindungsregeln oder die Speicherung hydrologisch wichtiger Punkte wie Quellen, Zusammenflüsse oder Senken in die Hierarchisierung eingebunden.

Prinzipiell ist es möglich, ein auf dieser Basis arbeitendes, hierarchiebasiertes Fließgewässer- und Einzugsgebietsmanagement-System sowohl lokal als auch inter- oder intranetbasiert in eine existierende Verwaltungsstruktur einzugliedern. Auf Grund der Kopplung und Struktur von Geoinformationssystem und Datenbank ergaben sich, über die Möglichkeiten der hierarchischen Abfragen hinaus, für Bearbeiter und Administration folgende Vorteile:

- Verbessertes Verständnis des Gebietes und seiner Funktionsweise. Der Benutzer ist in der Lage, einen raschen Überblick zu einem bestimmten Projekt und seinem geographischen und administrativen Kontext zu gewinnen.
- Verbesserte Kommunikation zwischen verschiedenen Bearbeitern, Abteilungen und Ämtern mit Entscheidungskompetenzen im Bereich der Fließgewässer.
- Verbesserte Kohärenz zwischen den Informationen von Daten einheitlicher Art durch Normalisierung bzw. Standardisierung.
- Höhere Zuverlässigkeit bezüglich der Speicherung von Informationen und Erhöhung ihrer Dauerhaftigkeit [BLASCHKE 1997].

Der allgemeine Gewinn durch eine solche Applikation besteht also idealerweise in einer optimierten Bearbeitung und Verwaltung von Gewässern und ihrer Daten. Entscheidungen können mit verbesserter Kenntnis der laufenden Projekte und der geographischen Umgebung getroffen werden. Denkbar ist z. B., die Resultate jeder Analyse, Messung und Berechnung zu integrieren und auf den verschiedenen Management-Ebenen miteinander zu vergleichen, um auf die hauptsächlichen Auswirkungen schließen zu können. Diese Synthese ist von grundlegender Wichtigkeit, um Entscheidungen zu finden, welche die Rahmenbedingungen optimal einbinden. So könnte z. B. bei einem Bauvorhaben am Gewässer der Verantwortliche für die Gebietsplanung auf die Existenz einer Einleitung am rechten Ufer hinweisen, und der Verantwortliche für die Wasserqualität auf eine empfindliche Zone flussabwärts. Für ein effizientes Projektmanagement müssten diese beiden Zwänge von Anfang an in der Konzeption der Verbauung berücksichtigt werden. Bei der Effektivierung solcher Vorgänge kann die hierarchische Basis an Metadaten zum Gewässernetz von entscheidender Bedeutung sein.

Solche Applikationen stellen somit auch Instrumente für die Entscheidungsfindung (Decision Support System, DSS) dar und sollen den Entscheidungsträgern zur Unterstützung dienen. In der Kombination von räumlicher Analyse im GIS, Datenbankabfragen und e-

ventuell Resultaten von Simulationsmodellen wird ein DSS zu einer wesentlichen Hilfe im Entscheidungsfindungsprozess einer Verwaltung [GIUPPONI ET AL. 2002]. Dabei soll es in der Lage sein, eine Vielzahl geographischer Informationsschichten (Layer, Themen) sowie unterschiedliche Ausmaße der Dokumentation (Gebiete mit keinen, wenigen und vielen Daten) zu berücksichtigen. Um dieser Heterogenität der Situationen gerecht zu werden, muss das DSS sehr flexibel sein bezüglich der räumlichen Darstellung und der simulierten Prozesse – eine Voraussetzung, die durch die hierarchische Datengrundlage unterstützt wird. Das System ermöglicht die Extraktion derjenigen synthetischen Parameter, welche für das Management der Fließgewässer entscheidend sind. Die Entscheidungsfällungen beruhen also nicht auf den rohen Werten einer Messreihe, Formel oder Simulation, sondern auf der Gegenüberstellung dieser Werte mit der geographischen Situation. Die räumliche Analyse mithilfe des GIS, einer Tabelle oder eines Hierarchieexplorers, gibt den Verantwortlichen eine gesamtheitliche und ergonomische Sicht auf die verwaltete geographische Einheit. Die vorgestellten Applikationen beinhalten daher unterschiedliche Sichten auf die Daten. Dies sind im einzelnen:

- Eine **Kartensicht** mit den Rauminformationen (diese wurde auch unter ArcSDE getestet).
- Eine **Tabellensicht**, die direkt an die Datenbank gebunden und unabhängig von den Geometriedaten ist. Hierarchische Abfragen können also sowohl für die Karte als auch die Tabelle unabhängig voneinander gestellt werden. In der Regel erfolgt jedoch die Abfrage auf die Datenbank (via SQL) zuerst. Das hierarchisch eingeschränkte Resultat kann dann auch in der Karte dargestellt werden. Ggf. wird der SQL-Abfrage-Ausdruck in einen Filter (entweder für das Shapefile oder die Query auf die Datenbank) umgewandelt. Neben der Effizienz ist damit auch immer die Übereinstimmung der Selektion der Datensätze aus der Datenbank mit den Geometriedaten gewährleistet.
- Eine **Hierarchiesicht.** Diese wurde in verschiedenen Applikationen unterschiedlich implementiert: innerhalb des HEIS (Hierarchiebasiertes Einzugsgebiets-Informations- und Monitoringsystem) mithilfe eines sogenannten Treeview, in der internetfähigen Anwendung mittels HTML bzw. einer rekursiven PHP-Funktion. Auch diese ist mit den beiden zuvor genannten Komponenten verlinkt.
- Eine **Diagrammsicht** für abwärts gerichtete und verbindende Fragestellungen.
- Eine **Reportsicht**, in der für den betrachteten Ausschnitt aus der Datengrundlage statistische und andere (z. B. Bilanzen) Informationen dargestellt werden.

Auf diese Weise ist es möglich, von der Tabelle, der Karte oder der Hierarchiesicht aus mithilfe der integrierten Werkzeugsammlung hierarchische Abfragen anzustoßen. Wird also z. B. die Verbindung zweier Abschnitte erfragt, so kann diese in der Karte, der Tabelle, der Hierarchiesicht (Selektion innerhalb des Baumes), als Laufentwicklung im Diagramm und als Report in Form von Text auf einen Blick dargestellt werden. Durch die Verbindung der einzelnen Informationsträger wird die Arbeit mit den Gewässerdaten intuitiver und ergonomischer. Die Vorteile liegen vor allem in der Visualisierung von Zusammenhängen auf Basis der hierarchischen Beziehungen mithilfe der unterschiedlichen Programmkomponenten und der damit verbundenen Optimierung wassergütewirtschaftlicher Bilanzierungs-, Überwachungs- und Modellierungsprozesse – natürlich in Abhängigkeit von der jeweiligen Fragestellung, den verfügbaren Ausgangsdaten und Modellkopplungen.

Perspektivisch besteht darüber hinaus die Möglichkeit, verschiedene Modelle der räumlichen oder zeitlichen Analyse oder Simulationsmodelle dynamischer, natürlicher und anthropogener Phänomene anzubinden und den Datenbestand mit den Resultaten zu er-

gänzen. Beispielsweise wäre eine Kopplung des Netzwerkes mit hydrologischen Modellen zur Beschreibung des flächendifferenzierten Wasser- und Stoffhaushaltes im Einzugsgebiet (Niederschlags-Abfluss-Modelle, Wasserhaushaltsmodelle) denkbar, sofern die Datengrundlagen wie Landnutzung, Niederschlagsmessreihe, Grundwasserflurabstand, Einzugsgebiets- und Fließgewässergeometrie, Bodeninformationen, Höhen- und Reliefinformationen usw. vorhanden sind. So ist es vorstellbar, das entsprechende hydrologische Modell (z. B. ArcEGMO [PFÜTZNER ET AL. 1999], SWAT oder WASIM-ETH) mit einer User-Interaktion (z. B. Klick in die Karte) bzw. bei Veränderungen der Datengrundlage (beispielsweise bei Hinzukommen oder Wegfallen eines Messpunktes) für die Oberlieger eines Abschnittes oder Messpunktes zu starten.

Diese Arbeit soll einen Beitrag leisten, die wachsende Menge anfallender Daten zum Fließgewässereinzugsgebietsmanagement effizienter zu verarbeiten und nutzbar zu machen. **Abb. 1** gibt einen Überblick zu den einzelnen Entwicklungen. Besonders in Hinblick auf die sich aus der EU-Wasserrahmenrichtlinie ergebenden Anforderungen sind solche Werkzeuge zur Erstellung und Nutzung logischer Netzwerke wichtige Hilfsmittel für Behörden, Institutionen und bearbeitende Büros sowie zur Veröffentlichung umfassender Hintergrundinformationen über das Internet und sollten verstärkt, auch in bereits vorhandenen Managementstrukturen, wie beispielsweise das UMWELT-INFORMATIONS-SYSTEM (UIS) des Landes Sachsen Anhalt, eingesetzt werden.

Anhang

Literatur- und Quellenverzeichnis

[ACM 2002] ACM (2002): Programming Perl - An interview with Larry Wall. URL: http://www.acm.org/crossroads/xrds1-2/lwall.html [Stand: 20. 8. 2002].

[AHUJA ET AL 1993] AHUJA, R. K., T. L. MAGNANTI and J. B. ORLIN (1993): Network Flows: Theory, Algorithms, and Applications Prentice Hall, New York.

[APACHE 2003] Apache (2003): The Apache Software Foundation: http://www.apache.org [Stand 20.03.03].

[BAILEY & GATRELL 1995] BAILEY, T. and A. GATRELL (1995): Interactive Spatial Data Analysis. Longman, Essex.

[BARRINGER & LILBURNE 1997] BARRINGER, J. and L. LILBURNE (1997): An Evaluation of Digital Elevation Models for Upgrading New Zealand Land Resource Inventory Slope Data. Second annual conference of Geo Computation '97 & SIRC '97. Otago (New Zealand).

[BARROW 1998] BARROW C. (1998): River basin development planning and management: A critical review. World Development 26 (1), o. O., 171 – 186.

[BARTHELME 1995] BARTHELME, N. (1995): Geoinformatik - Modelle, Strukturen, Funktionen. Berlin.

[BAUMGARTNER & LIEBSCHER 1996] BAUMGARTNER, A. und H. J. LIEBSCHER (1996): Allgemeine Hydrologie / Quantitative Hydrologie (Lehrbuch der Hydrologie Band 1). Berlin.

[BILL & FRITSCH 1991] BILL, R. und D. FRITSCH (1991): Grundlagen der Geo-Informationssysteme. Heidelberg.

[BILL 1996] BILL, R. (1996): Grundlagen der Geo-Informationssysteme, Band 2. Heidelberg.

[BLASCHKE 1997] BLASCHKE, TH. [Hrsg.] (1997): Landschaftsanalyse und -bewertung mit GIS - Methodische Untersuchungen zu Ökosystemforschung und Naturschutz am Beispiel der bayerischen Salzachauen. Deutsche Akademie für Landeskunde, Trier.

[BLÖCH 1999] BLÖCH, H. (1999): Europäische und internationale Normen zu Gewässerschutz und Wasserwirtschaft. In: Das Recht des Wassers in nationaler und internationaler Perspektive. Wien, S. 33 - 43.

[BLÖCH 2001] BLÖCH H. (2001): Europäische Ziele im Gewässerschutz. Auswirkungen der EU-Wasserrahmenrichtlinie auf Deutschland. In: KA Wasserwirtschaft, Abwasser, Abfall. Heft 2, Hennef, S. 168 – 172.

[BLONGEWICZ 2000] BLONGEWICZ, M. (2000): The development of an Integrated Catchment Modelling System using ArcView GIS. In: Proceeds of the European Conference on Advances in Flood Research Vol. 1. Potsdam.

[BOOCH 1994] BOOCH, G. (1994): Object-Oriented Design with applications. Redwood City (California).

[BURTON 1995] BURTON J. (1995): A framework for integrated river basin management. In: Water Science and Technology. Vol. 32 (5-6), o. O., p. 139 - 144.

[BÜYÜKSALIH & JACOBSEN 2002] BÜYÜKSALIH, G. and K. JACOBSEN (2002): Determination and Improvement of Digital Elevation Models Based on Moms-2p Imagery. Turkish-German Geodetic Days. Berlin.

[CAR & FRANK 1994] CAR, A. and A. U. FRANK (1994): General principles of hierarchical spatial reasoning - the case of wayfinding. In: Proceedings of Sixth Int. Symposium on Spatial Data Handling SDH '94. Waugh T. C. and R. G. Healey [Hrsg.], IGU, Vol. 2, o. O., p. 646 - 664.

[CAR & TAYLOR 1999] CAR, A. and G. TAYLOR (1999): Parameters for spatial hierarchization of networks. The 11th European Colloquium on Quantitative and Theoretical Geography. Durham (England).

[CHAWATHE 1994] CHAWATHE, S. (1994): On Index Selection Schemes for Nested Object Hierarchies. Proc. of the Conference of Very Large Databases, Santiago de Chile.

[CHENG ET AL. 2001] CHENG, Q. , H. RUSSELL, D. SHARPE, F. KENNY, and P. QIN (2001): GIS-based statistical and fractal/multifractal analysis of surface stream patterns in the Oak Ridges Moraine. Computers & Geosciences. Vol. 27, o. O., p. 513 - 526.

[CODD 1970] CODD, E. F. (1970): A relational Model of Data for large shared Data Banks. In: Communications of the ACM. New York, Vol. 13 No. 6, o. O., 377 - 387.

[DARWIN 1859] DARWIN, Ch. (1859): On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life. London.

[DAVIS 1998] DAVIS, P. (1998): PHP-Script zur Erzeugung eines HTML-basierten Diagramms. URL: http://www.pobox.com/~pdavis [Stand 15.06.2002].

[DIEKKRÜGER 1999] DIEKKRÜGER, B. (1999): Regionalisierung von Wasserquantität und -qualität - Konzepte und Methoden. In: STEINHARDT, U. & M. VOLK [Hrsg.]: Regionalisierung in der Landschaftsökologie. Leipzig, 67 - 78.

[DIESTEL & KÜHN 2003] DIESTEL R. and D. KÜHN [Hrsg.] (2003): Topological Paths,Cycles and Spanning Trees in Infinite Graphs. Mathematisches Seminar Universität Hamburg, URL: http://www.math.uni-hamburg.de/home/diestel/papers/TST.pdf [Stand: 1/11/2001].

[DIJKSTRA 1959] DIJKSTRA, E. W. (1959): A note on two problems in connection with graphs. Numerische Mathematik. (1) 269 - 271 In: Charles B., D. P. McVey, B. C. Clements, M. Andrew and J. Parkes (1999): Worldwide Aeronautical Route Planner. Eugene (Oregon).

[DJAFRI ET AL. 2002] DJAFRI, N., A. A. A. FERNANDES, N. W. PATON and T. GRIFFITHS (2002): Spatio-Temporal Evolution: Querying Patterns of Change in Spatio-Temporal Databases, Proc. 10th ACM Int. Symposium on Advances in Geographic Information Systems (ACM-GIS), New York, p. 35 - 41.

[DODDS & ROTHMAN 2000] DODDS, P. S. and D. H. ROTHMAN (2000): Scaling, Universality, and Geomorphology. Annu. Rev. Earth Planet. Sci. O. O., p. 28.

[DÖRHÖFER & DIBBERN 2001] DÖRHÖFER G. und B. DIBBERN (2001): Die Umsetzung der europäischen Wasserrahmenrichtlinie - eine Herausforderung an die Geowissenschaften. In: Instrumentarien zur nachhaltigen Grundwasserbewirtschaftung. Braunschweig.

[DUTTMANN, & HERZIG 2002] DUTTMANN, R. und A. HERZIG (2002): Vorhersage von Boden- und Gewässerbelastungen mit einem GIS-basierten Prognosesystem. In: MAYR, A., M. MEURER und J. VOGT [Hrsg.]: Stadt und Region, Dynamik von Lebenswelten. Tagungsbericht und wissenschaftliche Abhandlungen. 53. Dt. Geographentag, Leipzig 2001.

[DVWK 1993] Deutsche Vereinigung für Wasserwirtschaft, Abwasser und Abfall (DVWK) (1993): Aussagekraft von Gewässergüteparametern in Fließgewässern - Teil I: Allgemeine Kenngrößen, Nährstoffe, Spurenstoffe und anorganische Schadstoffe, biologische Kenngrößen. Merkblatt 227, o. O., S. 3.

[DYCK & PESCHKE 1995] DYCK, S. und G. PESCHKE (1995): Grundlagen der Hydrologie. Berlin.

[ECO 1993] ECO, U. (1993): Wie man eine wissenschaftliche Arbeit schreibt. Stuttgart.

[EGENHOFER 1997] EGENHOFER, M. J. (1997): Query Processing in Spatial-Query-by-Sketch. Journal of Visual Languages and Computing. Vol. 8, o. O., p. 403 - 424.

[ERWIG 1994] ERWIG, M. (1994): Graphs in Spatial Databases. Doktorarbeit, Fernuniversität Hagen.

[ESRI 2000] ESRI - Environmental System Research Institute (2000): Shapefile Technical Description.
URL: http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf [Stand 10.03.2001]

[ESRI 2001] ESRI - Environmental System Research Institute [Hrsg.] (2001): Getting Started with SDE.
URL: http://www.geography.ccsu.edu/harmonj/g478/getstart.pdf [Stand 10.03.2001]

[FERNANDES 1987] FERNANDES, D. (1987): Inventário das estacóes fluviomêtricas, Ministerio das Minas e Energia, Departamento Nacional de Aguas e Energia Eléctrica, Divisaó de Controle de Recursos Hydricos, SGAN Q. 603 N Mod. J, Brasilia-DF-Brazil, CEP: 70830-030 In: K. L. VERDIN and J. P. VERDIN (2000): A topological system for delineation and codification of the Earth's river basins. Journal of Hydrology. Vol. 218, o. O., p. 1 - 12.

[FINDEISEN 1990] FINDEISEN, D. (1990): Datenstruktur und Abfragesprachen für raumbezogene Informationen. In: Schriftenreihe des Inst. f. Kartographie und Topographie der Rhein. Friedr. - Wilh. Universität Bonn, Heft 19, Bonn.

[FLÜGEL & STAUDENRAUSCH 1999] FLÜGEL, W. - A. & H. STAUDENRAUSCH (1999): Hydrological Network Modelling using GIS for supporting Integrated Water Resources Management. Proceedings of the International Congress on Modelling and Simulation, Vol.4, p. 1087 - 1092, OXLEY, L. and F. SCRIMGEOUR [Eds.], University of Waikato, Hamilton (New Zealand).

[FRIEDRICH & FRÜHAUF 2002] FRIEDRICH, K. und M. FRÜHAUF (2002): Halle und sein Umland: Geographischer Exkursionsführer. Halle (Saale).

[FRIEDRICH 2002] FRIEDRICH, G. A. (2002): Vergleich von Internet-GIS-Applikationen dargestellt am Beispiel verschiedener geowissenschaftlicher Nutzeranforderungen. Diplomarbeit, Inst. f. Geographie, FB Geowissenschaften, MLU, Halle (Saale).

[FRIGIONI ET AL. 2000] FRIGIONI, D., A. MARCHETTI - SPACCAMELA and U. NANNI (2000): Fully Dynamic Algorithms for Maintaining Shortest Paths Trees, Journal of Algorithms 34, o. O., p. 251 - 281.

[FRITSCH & ANDERS 1996] FRITSCH D. und K.-H. ANDERS (1996): Objektorientierte Konzepte in Geo-Informationssystemen, GIS, Jahrgang 9, Heft 2, S. 2 ff., o.O.

[FUHRMANN 2001] FUHRMANN, P. (2001): Konsequenzen aus der EU-Wasserrahmenrichtlinie für die Wasserwirtschaft in Deutschland. In: KA Wasserwirtschaft, Abwasser, Abfall. Hennef, S. 183 - 185.

[GABRIEL & RÖHRS 1995] GABRIEL R. & H. - P. RÖHRS (1995): Datenbanksysteme. Berlin.

[GANTER 2001] GANTER, J. (2001): ArcView and MapObjects: An Architectural Comparison for Application Developers. Sandia National Laboratories.
URL: http://wilds.sandia.gov/a/avmo [Stand 03. 02. 2003]

[GEOTOOLS 2003] Geotools (2003): URL: http://www.geotools.org [Stand 05. 05. 2003].

[GEPPERT 1999] GEPPERT, A. (1999): Objektorientierte Datenbanksysteme: ein Praktikum. Heidelberg.

[GIUPPONI ET AL. 2002] GIUPPONI, C., J. MYSIAK, A. FASSIO and V. COGAN (2002): Towards a spatial decision support system for water resource management: MULINO-DSS 1st release-. 5th AGILE Conference on Geographic Information Science, Palma (Balearic Islands, Spain).

[GOULTER & FORREST 1987] GOULTER, I. and D. FORREST (1987): Use of geographical information systems (GIS) in river basin management. Water Science and Technology. Vol. 19, o.O., p. 81-86.

[GRIFFITHS ET AL. 2002] GRIFFITHS, T., A. FERNANDES, N. W. PATON, S. - H. JEONG, N. DJAFRI, K. MASON, B. HUANG, M. WORBOYS (2002): Tripod: A Spatio-Historical Object Database System, Mining Spatio-Temporal Information Systems, The Kluwer International Series in Engineering and Computer Science: Volume 699, LADNER, R., K. SHAW and M. ABDELGUERFI [Eds.], o. O., p. 127-146.

[GUTING 2000] GUTING, R. (2000): GraphDB: A Data Model and Query Language for Graphs in Databases. Informatik Berichte. Heft 155, S 2, Fernuniversitat Hagen.

[HABERLANDT ET AL. 2000] HABERLANDT, U., B. KLÖCKING, V. KRYSANOVA and A. BECKER (2000): Regionalization of dynamically simulated flow components for large scale water resources assessment - a case study in the Elbe River Basin (submitted to J. Hydrol.).

[HAHN ET AL. 2000] HAHN H. K., SELLE D, C. J. G. EVERTSZ und H. - O. PEITGEN (2000): Strukturanalyse und Morphometrie interagierender Gefäßsysteme am Beispiel der menschlichen Leber. Workshoptopic: Vortrag: Quantifizierung von Bildinhalten. URL: http://www.imse.med.tu-muenchen.de/mi/bvm2000/program/ abstracts/w088.html [Stand: 1/11/2001].

[HEUER 1997] HEUER, A. (1997): Objektorientierte Datenbanken, Konzepte, Modelle, Standards und Systeme. Bonn.

[HEUVELINK 1993] HEUVELINK, G. B. M. (1993): Error propagation in quantitative spatial modelling applications in Geographical Information Systems. PhD-thesis Department of Physical Geography, University of Utrecht. Utrecht.

[HORTON 1945] HORTON, R. E. (1945): Erosional Development of Streams and their drainage basins: Hydro-physical approach to quantitative morphology. In: Bulletin of the Geological Society of America, 56. Denver. In: MARCINEK, J. und E. ROSENKRANZ (1996): Das Wasser der Erde - Eine geographische Meeres- und Gewässerkunde. Gotha.

[INTERWIES & KRAEMER 2001] INTERWIES, E. und R. A. KRAEMER (2001):Ökonomische Anforderungen der EU-Wasserrahmenrichtlinie. Analyse der relevanten Regelungen und erste Schritte zur Umsetzung. Endbericht an das Umweltbundesamt (UBA). In Abstimmung mit dem Bundesministerium für Umwelt, Naturschutz und Reaktorsicherheit (BMU) und dem Expertengremium "Umweltökonomie" der Länderarbeitsgemeinschaft Wasser (LAWA). Institut für Internationale und Europäische Umweltpolitik [Hrsg.], Berlin.

[JANKOWSKI 1995] JANKOWSKI, G. (1995): Zur Geschichte des Mansfelder Kupferschieferbergbaus. Clausthal-Zellerfeld.

[JENNINGS 2001] JENNINGS, R. (2001): Datenbanken mit Visual Basic 6. Markt und Technik, München.

[JENSON & DOMINGUE 1988] JENSON, S. K. and J. O. DOMINGUE (1988): Extracting topographic structures from digital elevation data for geographic information system analysis. Photogrammetric Engineering and Remote Sensing. Vol. 54(11), o. O., p.1593 - 1600.

[JING ET AL. 1998] JING, N., Y. - W. HUANG and E. A. RUNDENSTEINER (1998): Hierarchical Encoded Path Views for Path Query Processing: An Optimal Model and its Performance Evaluation. IEEE Transactions on Knowledge and Data Engineering. Vol. 10(3), o. O., p. 409 - 432.

[KARACAPILIDIS ET AL. 1997] KARACAPILIDIS, N., D. PAPADIAS and M. EGENHOFER (1995): Collaborative Spatial Decision Making with Qualitative Constraints. Artificial Intelligence Research Division [Hrsg.], Sankt Augustin.

[KEMPER & WALLRATH 1987] KEMPER A. and M. WALLRATH (1987): An analysis of geometric modeling in database systems. Computing Surveys, Vol. 19, No. 1, o. O., p. 47 - 91.

[KNITZSCHKE 1995] KNITZSCHKE, G. (1995): Metall- und Produktionsbilanz für die Kupferschieferlagerstätte im östlichen Harzvorland. In: JANKOWSKI, G. [Hrsg.] (1995): Zur Geschichte des Mansfelder Kupferschieferbergbaus, Clausthal-Zellerfeld. S. 270 - 284.

[KOSCHITZKI 1999] KOSCHITZKI, TH. (1999): Untersuchung zur Automatisierung und Optimierung von Gewässergüteüberwachung und Gütedatenmanagement mithilfe der Erweiterung eines objektorientierten GIS, dargestellt am Beispiel des Fließgewässersystems der SALZA. Diplomarbeit, Inst. f. Geographie, FB Geowissenschaften, MLU, Halle (Saale).

[KRATZ & SUHLIG 1997] KRATZ, R. und F. SUHLIG (1997): GIS in Naturschutz, Forschung, Planung, Praxis. Magdeburg.

[KRUMBIEGEL & SCHWAB 1974] KRUMBIEGEL, G. und M. SCHWAB (1974): Saalestadt Halle und Umgebung - Ein geologischer Führer. Teil 1: Geologische Grundlagen. Halle (Saale).

[KUMLER 1994] KUMLER, M.P. (1994): An intensive comparison of traiangulated irregular networks (TINs) and Digital Elevation Models (DEMs). Cartographica. Monograph. Vol. 31(2), o. O.

[LANG & LOCKEMANN 1995] LANG S. & P. LOCKEMANN [1995]: Datenbankeinsatz. Berlin.

[LAU 1997] Landesamt für Umweltschutz (LAU) Sachsen-Anhalt (1997): Gewässergütebericht Sachsen-Anhalt 1996. Halle (Saale).

[LAWA 1978] Länderarbeitsgemeinschaft Wasser (LAWA) [Hrsg.] (1978): Gewässereinzugsgebiete mit Unterteilung und Kennzeichnung nach LAWA Schlüssel.

[LAWA 2000] Länderarbeitsgemeinschaft Wasser (LAWA) [Hrsg.] (2000): Forderungen der Wasserwirtschaft für eine fortschrittliche Gewässerschutzpolitik.

[LAWA 2002] Länderarbeitsgemeinschaft Wasser (LAWA) [Hrsg.] (2002): Handlungskonzept zur Umsetzung der Wasserrahmenrichtlinie.

[LEHNER 1998] LEHNER, B.: Automated Generation of a Topologic River Network System within a GIS. In: STROBL, J. und F.DOLLINGER (1998): Angewandte Geographische Informationsverarbeitung. Beiträge zum AGIT-Symposium Salzburg. Heidelberg, S. 324 - 334.

[LIEBIG 1997] LIEBIG, W. (1997): Desktop-GIS mit ArcView. Heidelberg

[LIME 2001] LIME, S. (2001): Mapplet: Java Applet zur Rückgabe eines wählbaren Kartenausschnittes. URL: http://rpms.trider.uk/php/sources [Stand 12.09.2002].

[LOBUE 2002] LOBUE (2002): Terms and Concepts of Business Problems. URL: http://www.lobue.com/enterprise_evolution/knowledge_hierarchy.html. [Stand 21.11.2002].

[LOCKMAN 1998] LOCKMAN, D. (1998): Oracle 8 Datenbankentwicklung, München.

[LOUCKS 2002] LOUCKS, D. P. (2002): Sustainable Water Resources Management. International Water Resources Association (IWRA), Water International, Volume 25, p. 3 – 10.

[MAHESHWARI & ZEH 2001] MAHESHWARI, A. and N. ZEH [Hrsg.] (2001): Optimal Algorithms for Planar Graphs Using Separators. School of Computer Science, Carleton University, Ottawa.

[MAIER & BACHMANN 1997] MAIER, A. and P. BACHMANN (1997): Der Einsatz objektorientierter Datenbanksysteme in der Praxis: ein Triathlon. In: DITTRICH, K. und A. GEPPERT (Hrsg.): Datenbanksysteme für Büro, Technik und Wissenschaft. Berlin, S. 17 - 33.

[MARCINEK & ROSENKRANZ 1989] MARCINEK , J. und E. ROSENKRANZ (1996): Das Wasser der Erde - Eine geographische Meeres- und Gewässerkunde. Gotha.

[MCKINNEY ET AL. 1997] MCKINNEY , D. C., X. CAI and D. R. MAIDMENT (1997): A prototype GIS-based Decision Support System for River Basin Management. Paper presented at the 17th Int. ESRI User Conference. San Diego (USA).

[MEHLHORN 1986] MEHLHORN K. (1986): Datenstrukturen und effiziente Algorithmen. Bd. 1, Stuttgart.

[MEYERS 1992] MEYERS, S. (1992): Effektiv C++ programmieren. Bonn.

[MIDEWA 2003] MIDEWA Wasserversorgungsgesellschaft in Mitteldeutschland mbH (2003): mündliche Mitteilung vom 10.07.2003

[M. MÜLLER 2001] MÜLLER, M. (2001): Geodaten im Risikomanagement - Zonierungssysteme bei der Deutschen Versicherungswirtschaft (GDV). Versicherungskammer Bayern (Hrsg.), München.

[MYSQL 2002] MySQL (2002), Technical Information. URL: http://www.mysql.com/information [Stand: 29. 10. 2002].

[OGC 2001] OGC - OpenGIS® Consortium, Inc. (2001): URL: http://www.opengis.org [Stand 20.10.01].

[PAIVA & EGENHOFER 1997] DE CARVALHO PAIVA, J. A. and M. EGENHOFER (1996): Robust Inference of the Flow Direction in River Networks, Algorithmica (in press) URL: http://www.spatial.maine.edu/~max/FlowDirectionInference.pdf [Stand: 04. 3. 2002].

[PAIVA 1998] DE CARVALHO PAIVA, J. A. (1998): Topological Equivalence and Similarity in Multi-Representation Geographic Databases. Doctor Thesis, University of Maine, Augusta.

[PFÜTZNER 1996] PFÜTZNER, B. (1996): Hydrologisches Modell Salza – Teilprojekt „Hydrologische Grundlagenuntersuchungen im Einzugsgebiet der Salza“. Berlin.

[PFÜTZNER ET AL. 1999] PFÜTZNER, B., LAHMER, W., BECKER, A. und B. KLÖCKING (1999): ArcEGMO – GIS-gestützte hydrologische Modellierung. Programmdokumentation, Potsdam-Institut für Klimafolgenforschung (PIK), Potsdam.

[RENNER 2001] RENNER, J. (2001): Die EU-Wasserrahmenrichtlinie: verbesserter Gewässerschutz oder nur eine neue "Berichts- und Verwaltungsvorschrift"? In: KA Wasserwirtschaft, Abwasser, Abfall. Hennef, 2/2001 S. 147.

[RINALDO & RODRIGUEZ-ITURBE 2001] RINALDO, A. and I. RODRIGUEZ-ITURBE (2001): Channel networks. Annual Review of Earth and Planetary Sciences. Vol. 26, o. O., 289 - 327.

[ROCHE 1968] ROCHE, M. (1968): Traitement automatique de donné es hydromé tri-queset des donné es pluviomé triques au service hydrologique de l’ORSTOM, Cahiers de l’ORSTOM. In: VERDIN, K.L. and J.P. VERDIN (2000): A topological system for delineation and codification of the Earth’s river basins. Journal of Hydrology. Vol. 218, p. 1 - 12.

[RÖSCH 1998] RÖSCH, N. (1998): Topologische Beziehungen in Geo-Informationssystemen. Dissertation, Universität Fridericiana zu Karlsruhe.

[ROSEMANN & VERDAL 1970] ROSEMANN, H. J. und J. VERDAL (1970): Das Kalinin-Miljukow-Verfahren zur Berechnung des Ablaufs von Hochwasserwellen. Schriftenreihe der Bayerischen Landesstelle für Gewässerkunde, Heft 6, München.

[SAURER & BEHR 1997] SAURER, H. und F. J. BEHR (1997): Geographische Informationssysteme - eine Einführung. Darmstadt.

[SCHANZE 1998] SCHANZE J. (1998): Integratives Sanierungs- und Entwicklungskonzept für das Einzugsgebiet der Salza. 3. Zwischenbericht. Halle (unveröffentlicht).

[SCHMIDT & FRÜHAUF 1997] SCHMIDT, G. und M. FRÜHAUF (1997): Untersuchungen zur Bedeutung der Schwermetallemissionen aus den Halden des Mansfelder Kupferschieferbergbaus als Ursache von Boden- und Fließgewässerbelastungen. In: Hercynia. Leipzig, Bd. 30, S. 177 – 193

[SCHMIDT 1984] SCHMIDT, K. - H. (1984): Der Fluss und sein Einzugsgebiet. Hydrogeographische Forschungspraxis. Wiesbaden.

[SCHMIDT ET AL. 1992] SCHMIDT, G., ZIERDT, M und M. FRÜHAUF (1992): Die wassergebundene Schwermetallemission aus Halden des Mansfelder Kupferschieferbergbaus in das Vorflutsystem des Süßen Sees. In: Geoökodynamik. Bd. XIII, Heft 2, Bensheim, S. 153 - 172.

[SCHORGHOFER & DANIEL 1998] SCHORGHOFER, N. and H. R. DANIEL (1998): Causal relations between topographic slope and Drainage area. Geophysical research letters. Cambridge.

[SCHRECK 1996] SCHRECK, P. (1996): Zur Mobilisierung und Verbreitung von Schadstoffen aus den Halden des Mansfelder Kupferschieferbergbaus im Mansfelder Land. In: Bergbau und Umweltgeschichte in Mitteldeutschland, Sammelband zum Kolloquium an der MLU. Halle (Saale), S. 111 - 116.

[SCHRÖDER 1986] SCHRÖDER, H. (1986): Allgemein-geographische Charakteristik der natürlichen Verhältnisse des südöstlichen Harzvorlandes. In: Hercynia. Bd. 23, Heft 1, Leipzig, S. 1 - 14.

[SCHULZE 2000] SCHULZE, H. (2000): Lexikon Computerwissen. Hamburg.

[SEABER ET AL. 1987] SEABER, P., KAPINOS, F. and G. Knapp (1987): Hydrologic Unit Maps. USGS Water Supply Paper 2294, o. O., p. 63.

[SEDGEWICK 1992] SEDGEWICK, A. (1996): Algorithmen in C. Berlin.

[SHEKHAR, FETTERER & GOYAL 1993] SHEKHAR, S., FETTERER, A. and B. GOYAL (1997): Materialization Trade-Offs in Hierarchical Shortest Path Algorithms. In: Scholl, M. and A. Voisard [Hrsg.]: Advances in Spatial Databases. Proceedings of the 5th International Symposium, SSD '97. Lecture Notes in Computer Science. Vol. 1262, p. 94 - 111, Berlin.

[SHREVE 1966] SHREVE, R. L. (1967): Statistical law of stream numbers. In: Journ. Geol., Bd. 74, o. O., S. 17 – 37.

[SIMONOVIC 1996] SIMONOVIC, S. P. (1996). Decision Support Systems for Sustainable Management of Water Resources: 1. General Principles. In: Water International, Vol. 21, No. 4, o. O., p. 223 - 232.

[SINGER 1993] SINGER, CH. (1993): Relationale Datenbanken als Basis geographischer Datenbanken. GIS, Heft 6, o. O., S. 9 - 16.

[SMITH ET AL. 1991] SMITH, R. A., ALEXANDER R. B. and K. J. LANFEAR (1991): Stream Water Quality in the Conterminous United States - Status and Trends of Selected Indicators During the 1980's. National Water Summary 1990 - 91 - Stream Water Quality. U.S. Geological Survey Water-Supply Paper 2400. o. O.

[SMITH ET AL. 2002] SMITH, N. P., A. J. PULLAN and P. J. HUNTER (2002): An anatomically based Model of transient coronary Blood Flow in the Heart. Society for Industrial and Applied Mathematics, o. O. Vol.62, No.3, o. O., p. 990 - 1018.

[STAU 1998 A] Staatliches Amt für Umweltschutz (STAU) Halle [Hrsg.] (1998): Gewässergütebericht 1998. Halle (Saale).

[STAU 1998 B] Staatliches Amt für Umweltschutz (STAU) Halle [Hrsg.] (1998): Datengrundlagen zum Gewässergütebericht 1998. Halle (Saale).

[STRAHLER 1964] STRAHLER, A. N. (1964): Physical Geography, New York. In: MARCINEK, J. und E. ROSENKRANZ (1996): Das Wasser der Erde - Eine geographische Meeres- und Gewässerkunde. Gotha.

[SUKTHANKAR ET AL. 1992] SUKTHANKAR, R., HANCOCK, J. POMERLEAU and D. C. Thorpe (1992): A Simulation and Design System for Tactical Driving Algorithms. In: Proceedings of AI, Simulation and Planning in High Autonomy Systems. Carnegie Mellon University [Hrsg.], Oakland (California).

[THOMPSON & LAURINI 1992] THOMPSON, D. and R. LAURINI (1992). Fundamentals of Spatial Information Systems. Number 37 in APIC Series. Los Angeles.

[TIMPF & FRANK 2000] TIMPF, S. and A. U. FRANK (2000): Using Hierarchical Spatial Data Structures for Hierarchical Spatial Reasoning. In: HIRTLE, S. C. and A. U. FRANK [Hrsg.]: Spatial Information Theory - A Theoretical Basis for GIS (International Conference COSIT'97). Lecture Notes in Computer Science, Berlin, Vol. 1329, o. O., p. 69 - 83.

[UBERTINI 1999] UBERTINI, L. (1999): Hydrological sciences. Italian research activity (1995-1998) report to IAHS. Bollettino di Geofisica Teorica ed Applicata. Vol. 40, o. O., p. 83 - 91.

[UMN 2001] University of Minnesota (2001): UMN MapServer. Free Mapserver. URL: http://mapserver.gis.umn.edu [Stand 09.08.2001].

[USGS 2000] United States Geological Survey - USGS [Hrsg.] (2000): Evaluation of Water Quality Status in the Raritan River Basin, Water Years 1991 - 1997, o. O.

[VERDIN & VERDIN 2000] VERDIN, K. L. & J. P. VERDIN (2000): A topological system for delineation and codification of the earth's river basins. Journal of Hydrology, o.O., Vol. 218, p. 1-12,

[VERRAES 2001] VERRAES, W. (2001): Evolutie Der Vertebrata: Nota's bij de Cursus Morfologie en Systematiek der Dieren - Partim Vertebraten. Vorlesungsscript. Gent (Belgien).

[VOLKMANN 1996] VOLKMANN, L. (1996): Fundamente der Graphentheorie. Berlin.

[VOSSEN 1994] VOSSEN, G. (1994): Datenmodelle, Datenbanksprachen und Datenbank-Mangement-Systeme. Berlin.

[WAGNER & WILLHALM 2003] WAGNER D. and TH. WILLHALM (2003): Geometric Speed-Up Techniques for Finding Shortest Paths in Large Sparse Graphs. Konstanzer Schriften in Mathematik und Informatik. Nr. 183, Konstanz.

[WASY 1996] Gesellschaft für wasserwirtschaftliche Planung und Systemforschung mbH - WASY [Hrsg.] (1996): Bewirtschaftungsplan Salza - Teilprojekt. Untersuchungen zu Abflussverhältnissen und zum Stoffeintrag im Einzugsgebiet der Bösen Sieben. Berlin.

[WILHELM 1993] WILHELM, F. (1993): Grundlagen der Allgemeinen Hydrogeographie. 2. Auflage, Braunschweig.

[WILKINSON ET AL. 1986] WILKINSON, G. G. (1997): The Generalisation of Satellite-Derived Raster Thematic Maps for GIS input. Geo-Informations-Systeme 6 (5), 24 - 29 In: SAURER, H. und F.J. BEHR (1997): Geographische Informationssysteme - Eine Einführung. Darmstadt.

[WOHLRAB ET AL. 1992] WOHLRAB, B., H. ENRNSTBERGER, A. MEUSER UND V. SOKOLLEK (1992): Landschaftswasserhaushalt. Berlin.

[ZIERDT & SCHMIDT 1996] ZIERDT, M. und G. SCHMIDT (1996): Endbericht: Untersuchungen zum Abflußverhalten der Oberflächenwassergüte im Einzugsgebiet Wilder Graben - Teilleistung zum "Hydrologischen Einzugsgebietsmodell Süßer See". Halle (Saale).

[ZIETLOW 1993] ZIETLOW, M. (1993): Bäume und Schlangen im PC-Paradies. PC-Professionell, Berlin.

Verzeichnis der Abkürzungen

ANSI	American National Standard Institute
API	Application Program Interface – Schnittstelle zur Anwendungsprogrammierung
ASF	Apache Software Foundation
ATKIS	Amtliches Topographisch - Kartographisches Informationssystem
BLOB	Binary Large Object
CAD	Computer Aided Design
CGI	Common Gateway Interface
CLOB	Character Large Object
COM	Component Object Model
DB	Datenbank
DBMS	Datenbankmanagementsystem
DBS	Datenbanksystem
DDL	Data Definition Language
DGM	Digitales Geländemodell
DLM	Digitales Landschaftsmodell
DML	Data Manipulation Language
DMU	Danske Miljøundersøgelser Instituten = National Environmental Research Institute (NERI) in Selkeborg
DNAEE	Departamento Nacional de Aguas e Energia Eletrica
DNOS	Departamento Nacional de Obras de Saneamento
DVWK	Deutsche Vereinigung für Wasserwirtschaft, Abwasser und Abfall
EU-WRRL	EU-Wasserrahmenrichtlinie
EZG	Einzugsgebiet
FGWID	Fließgewässer-ID - eindeutige Kennung eines Fließgewässerabschnittes
FGWULID	Fließgewässer-Unterlieger-ID - FGWID des Unterliegers eines Fließgewässerabschnittes
FIFO	First In First Out - Zuerst eingelesen - zuerst ausgelesen
FTP	File Transfer Protocol
GIF	Graphics Interchange Format
GIS	Geographische Informationssystem
GLOWA	Projekt „Globaler Wandel des Wasserhaushaltes“
GRDC	Global Runoff Data Center
GUI	Graphical User Interface - Graphische Benutzerschnittstelle
HTML	Hypertext Markup Language - Kennzeichnungssprache zur Erstellung von Internetseiten
HUC	Hydrologic Unit Code
HUS	Hydrologic Unit System
ISO	International Organization for Standardization
IWRA	International Water Resources Association
JDBC	Java Database Connectivity
JPEG	Joint Photographic Interchange Group
LAU	Landesamtes für Umweltschutz (Halle)
LAWA	Länderarbeitsgemeinschaft Wasser
LLVermD	Landesamt für Landesvermessung und Datenverarbeitung
LUA	Landes-Umweltamt Brandenburg
MNDNR	Minnesota Department of Natural Ressources
MySQL	Opensource Datenbanksystem (relational)
NWIS	National Water Information System
ODBC	Open Database Connectivity
OGC	OpenGIS Consortium
OID	Objektidentität
OLE	Object Linking and Embedding
OODBMS	Objektorientierte Datenbanken

ORSTOM	heute “Institut français de Recherche scientifique pour le Développement en Coopération“ – IRD
PHP	Hypertext Preprocessor
PIK	Potsdam Institut für Klimafolgenforschung
Poet	objektorientierte Datenbank
PostGis	räumliche Erweiterung von PostGreSQL
PostGreSQL	objekt-relationale Datenbank
RDB	Räumliche Datenbank (generell)
RDBMS	relationales Datenbankmanagementsystem
SDE	räumliche Datenbank
SQL	Standard Query Language
SSL	Storage Structure Language
STAU	Staatliches Amt für Umweltschutz (Halle)
TCP/IP	Transmission Control Protocol/Internet Protocol
TEZG	Teileinzugsgebiet
TIN	Triangulated Irregular Network - Dreiecksvermaschung
TLUG	Thüringer Landesamt für Umwelt und Geologie
UL-OL-Beziehungen	Unter – Oberliegerbeziehungen, werden über die FGWID und FGWULID der Gewässerabschnitte ermittelt
UMN	University of Minnesota
USGS	US Geological Survey
XML	Extensible Markup Language - einfache Version des SGML - Standards für den Entwurf von HTML-Dokumenten

Codes

Anmerkung: Die folgenden Codeausschnitte wurden – soweit nicht anders benannt – eigenständig erstellt. Natürlich sollen und können hier nur exemplarisch einige, für das Verständnis von Problemlösungen interessante Bereiche eines Programms in komprimierter Form dargestellt werden. In diesem Zusammenhang marginale Komponenten wie z. B. Variablendeklarationen, Abfragestrukturen oder Fehlerbehandlung werden in der Regel ausgeblendet. Gegebenenfalls wurde auch Pseudocode eingesetzt, da es insgesamt ja nicht um die Abbildung der eigentlichen Programme geht (diese würden alleine für die Extension zum Präprozessing etwa 350 Seiten A4 in der vorliegenden Formatierung benötigen) und das Verständnis von Programmabläufen erhöht werden soll. Oft wird der Code jedoch in Auszügen aus der Originalversion mit erweiterter Kommentierung erklärt, um eine problemlose Umsetzung der Fragestellung in der betreffenden Umgebung zu unterstützen.

CODE 1: SCHEMA EINER IMAGINÄREN KLASSE INNERHALB EINES OODBMS

```
Class Abschnitt{
 Properties
  Dimension: Line;
  Name :String;
  FGWID, Unterlieger, Gebietsauslass:  Integer;
  Laenge, Q, Mng: Double;
  istQuelle: Boolean;
 Operations
  sucheNachbarn(FGWID): Datenstruktur;
  findeAlleOberlieger(FGWID): Datenstruktur;
  wegZumAuslass(FGWID, Gebietsauslass): Datenstruktur;
  sucheVerbindung(FGWID, Integer): Datenstruktur;
}
Class Quelle Inherit Abschnitt{
 Properties
  istQuelle: Set (Nachbarn) inverse is Nachbarn.count = 1;
 Operations
  gibHierarchischeEntfernungAn (Gebietsauslass);
  gibHierarchischeBreiteAn (Gebietsauslass);
}
```

CODE 2: AUSZUG AUS AVENUE-SUBROUTINE ZUR EXTRAKTION WICHTIGER PUNKTE AUS DEM GEWÄSSERNETZ

```
'zuerst die Listen definieren
shapes={} aussenpunkte={} springshapes={} pseudos={} bifs={} singleLines={} nodes={} tops={}
sinks={} vertices={} pseudolines={} centers={} starts={} ends={} pseudoLines={} verzweigun-
gen={} zusammenfluesse={} echteQuellen={}

for each r in theFtab.getSelection 'Schleife über alle selektierten Datensätze
  currentLine = theFtab.returnValue(theFtab.findField("Shape"), r) 'aktuelles Shape
  shapes.add(currentLine) 'zur Shapeliste hinzufügen
  centers.add(currentLine.returnCenter) 'Zentrumsliste
  Ml = currentLine.asMultipoint.asList 'auf Vertices der aktuellen Linien zugreifen
  if(Ml.count < 2)then continue end 'Linie ist nicht sauber
  vertices = vertices.merge(ML) 'Liste der Vertices
  start = Ml.get(0) 'Anfangspunkt
  ende = Ml.get(Ml.count-1) 'Endpunkt
  starts.add(start) 'Liste der Anfangspunkte
  ends.add(ende) 'Liste der Endpunkte
  theFtab.selectByPolyline(aktuelleLinie, #VTAB_SELTYPE_NEW) 'Nachbarn selektieren
  if(theFtab.getSelection.count = 1)then
    singleLines.add(aktuelleLinie) 'Liste der Einzellinien
  end
  for each p in {start, ende} 'Anfangs- und Endpunkt untersuchen
    if(Nodes.findByValue(p) <> (-1))then continue end 'Kontrolle auf Vorhandensein in Nodes
    nodes.add(p) 'zur Liste der Nodes hinzufügen
    founds = 0 'Variablen zurücksetzen
    Startes = 0
    Endes = 0
    for each k in theFtab.getSelection 'Untersuchung der Nachbar
      aline = theFtab.returnValue(theFtab.findField("Shape"), k)'aktuelle Linie dieser Selektion
      Ml = aline.asMultipoint.asList 'Vertices dieser Liste
      if(Ml.count < 2)then continue end 'Linie ist nicht sauber
      aStart = Ml.get(0) 'Anfangspunkt dieser Linie
      aEnde = Ml.get(Ml.count-1) 'Endpunkt dieser Linie
      if(p.intersects(aline))then 'Test auf Intersection
        founds = founds+1 'Festhalten des Tests
      end
      if(p.intersects(aStart))then Startes = Startes+1 end 'gefundene Anfangspunkte hochzählen
      if(p.intersects(aEnde))then Endes = Endes+1 end 'gefundene Endpunkte hochzählen
    end
    if(founds = 1)then 'muss Aussenpunkt sein
      aussenpunkte.add(p)  'zur Liste der Aussenpunkte hinzufügen
      springshapes.add(ashape) 'zur Liste der Quellen berührenden Linien hinzufügen
      if(p.intersects(Start))then 'muss Quelle sein
        echteQuellen.add(p) 'zur Liste der Quellen hinzufügen
      end
    end

    if(founds > 2)then bifs.add(p) end 'zur Liste der Bifurkationen hinzufügen
```

Codes

```
    if((Startes=founds)and(founds > 1))then Tops.add(p) end 'zur Liste der Tops hinzufügen
    if(Endes=founds)then Sinks.add(p) end 'zur Liste der Senken hinzufügen
    if(founds > 1)then
      if((Endes>Startes)and(Startes=1))then
        zusammenfluesse.add(p) 'zur Liste der Zusammenflüsse hinzufügen
      elseif(Endes<Startes)then
        Verzweigungen.add(p) 'zur Liste der Verzweigungen hinzufügen
      end
    end
    if(founds = 2)then 'muss Pseudonode sein
      pseudos.add(p) 'zur Liste der Pseudoknoten hinzufügen
      pseudolines.add(ashape)) 'zur Liste der zusammenzuführenden Linien hinzufügen
    end
  end
  theFtab.setSelection(oldsel) 'alte Selektion wiederherstellen
end
av.PurgeObjects 'Garbage Collector leeren
retlist = {aussenpunkte, bifs, pseudos, shapes, springshapes,
singleLines, nodes, vertices, pseudolines, centers, starts, ends, Tops, Sinks, Zusammenflues-
se, Verzweigungen, echteQuellen}) 'Rückgabeliste erstellen
return retlist 'Rückgabe
```

CODE 3 : VISUAL BASIC-FUNKTION - GIBT DIE SHAPES DER SELEKTIERTEN DATENSÄTZE IN EINER KARTE ALS COLLECTION (SAMMLUNG) ZURÜCK

```
Public Function Shapes ReturnFeatures(pFeatureLayer As IFeatureLayer) As Collection
  'Variable-Deklaration:
  Dim pMxDoc As ImxDocument 'Aktuelles Dokument
  Dim pEnumFeature As IEnumFeature 'Feature-Collection
  Dim pFeature As IFeature ' Feature
  Dim aCollection As New Collection 'Collection zur Aufnahme der räumlichen Objekte

  Set pMxDoc = ThisDocument 'Aktuelles Dokument in Variable schreiben
  Set pEnumFeature = pMxDoc.FocusMap.FeatureSelection 'Selektion in der Karte in Feature-Collection schreiben
  pEnumFeature.Reset 'an den Anfang der Collection gehen
  Set pFeature = pEnumFeature.Next 'erstes räumliches Objekt in der Selektion
  Do While Not pFeature Is Nothing 'alle Elemente der Selektion durchlaufen
    'Sicherstellen, dass nur Objekte des an die Funktion übergebenen Layers in Sammllung geschrieben werden
    If (pFeature.Class.AliasName = pFeatureLayer.FeatureClass.AliasName) Then
        aCollection.Add pFeature.Shape 'Shape des Objektes zur Collection hinzufügen
    End If
    Set pFeature = pEnumFeature.Next 'nächstes räumliches Objekt in der Selektion
  Loop
  Set Shapes_ReturnFeatures = aCollection 'Rückgabewert der Funktion
End Function
```

CODE 4: VISUAL BASIC-FUNKTION - GIBT DIE KREUZUNGEN DER SELEKTIERTEN DATENSÄTZE IN EINER KARTE ALS COLLECTION (SAMMLUNG) ZURÜCK

```
Public Function Shapes ReturnSpecificPoints ReturnBifs(aCollection As Collection,
pFeatureLayer As IFeatureLayer)'Übergeben werden die Nodes der selektierten Datensätze und das dazu gehörende Shapefile

Dim BifCollection as New Collection 'Sammlung für die Knoten anlegen
Dim BifPointCollection As IPointCollection 'Punktsammlung für die Knoten anlegen
Dim pFCur As IFeatureCursor 'Feature-Cursor zum selektiern innerhalb des aktiven Feature-Layers
Dim pFeature As IFeature 'Variable für räumliches Objekt
Dim aPoint As IPoint 'Punkt
Dim IntersectPolyline As IPolyline 'Polylinie
Dim i As Integer, c As Integer 'Integer-Variablen für Schleifendurchlauf

Set BifPointCollection = New Polyline 'Punktsammlung initiieren
For i = 1 To aCollection.Count 'übergebene Punktsammlung durchlaufen
   'Features in Layer mit Punkt selektieren; selektierte Elemente in Feature-Cursor schreiben
   Set pFCur = ReturnIntersectFeatureCursor (pFeatureLayer, aCollection(i))
   Set pFeature = pFCur.NextFeature 'räumliches Objekt (Feature) in Variable schreiben
   c = 0 'Integer-Variable auf 0 setzen
   Do While Not pFeature Is Nothing 'Anzahl der selektierten Elemente ermitteln
        c = c + 1
        Set pFeature = pFCur.NextFeature
   Loop
    If (c > 2) Then 'echter Knoten (mindestens von drei Linien berührt)
        Set aPoint = aCollection(i)
        If (pPointCollection.PointCount = 0) Then
```

Codes

```
          pPointCollection.AddPoint aPoint
        Else
          Set IntersectPolyline = BifPointCollection 'Testlinie erzeugen
          'Test, ob Punkt Testlinie berührt (siehe Funktion „Intersects“)
          If (Not Intersects(IntersectPolyline, aPoint)) Then
            BifPointCollection.AddPoint aPoint 'Punkt zur Punktsammlung hinzufügen
            BifCollection.Add aPoint 'Punkt zur Sammlung hinzufügen
          End If
        End If
    End If
  Next I
  Set Shapes_ReturnSpecificPoints_ReturnBifs = BifCollection 'Sammlung zurückgeben
End Function
```

CODE 5: VISUAL BASIC-FUNKTION - SELEKTIERT FEATURES IN LAYER MIT ÜBERGEBENEM PUNKT UND GIBT FEATURE-CURSOR ZURÜCK

```
Public Function ReturnIntersectFeatureCursor(pFeatureLayer As IFeatureLayer, selShape As
IGeometry) As IfeatureCursor
'Variablen-Deklaration:
Dim pFCur As IfeatureCursor ' Feature-Cursor anlegen
Dim pFC As IFeatureClass 'Feature-Class anlegen
  Set pFC = pFeatureLayer.FeatureClass 'Feature-Class des übergebenen Layers
  'räumliche Abfrage durchführen und in Feature-Cursor festhalten
  Dim pSF As IspatialFilter 'räumlichen Filter anlegen
  Set pSF = New SpatialFilter
  Set pSF.Geometry = selShape 'Selektions-Shape
  pSF.GeometryField = pFC.shapeFieldName 'räumlichen Filter einstellen
  pSF.SpatialRel = esriSpatialRelIntersects 'Filteroperation festlegen
  Set pFCur = pFC.Search(pSF, False) 'Feature-Cursor belegen
  Set SelectByShapeReturnIntersectFeatureCursor = pFCur 'Feature-Cursor an aufrufende Funktion zurück geben
End Function
```

CODE 6: VISUAL-BASIC-FUNKTION - ERHÄLT ZWEI SHAPES UND TESTET, OB SICH ZWEI RÄUMLICHE OBJEKTE BERÜHREN

```
Public Function Intersects(firstShape As IGeometry, secondShape As IGeometry) As Boolean
Dim pRelOp As IRelationalOperator 'räumlichen Operator festlegen
Set pRelOp = firstShape 'räumlichen Operator auf ersten übergebenen Shape setzen
Intersects = pRelOp.Touches(secondShape) 'true, wenn sich die beiden Objekte berühren, false, wenn nicht
End Function
```

CODE 7: AUSZUG AUS VISUAL-BASIC-FUNKTION ZUR ERMITTLUNG DES ZENTRUMS EINER LINIE

```
'theLine sowie die Prozenzahl kommen von der aufrufende Prozedur/Funktion
Public Function AlongLine(theLine As MapObjects2.Line perc As Double)
searchLength = (theLine.Length * perc) / 100 'prozentuale Länge
Set Vertices = returnVertices(aLine) 'Subroutine aufrufen
l = 0
For m = 1 To Vertices.Count - 1 'über alle Vertices
  dist = Vertices (m).DistanceTo(Vertices (m + 1)) 'Abstand zwischen Vertices
  l = l + dist 'Gesamtdistamz
  Set thePoint = Vertices(m) 'Punkt aktualisieren
  If (l >= searchLength) Then 'Nähe des gesuchten Bereiches
    quot = dist / (l - searchLength) 'Koordinaten ermitteln
    thePoint.x = Vertices(m).x + ((Vertices(m + 1).x - Vertices(m).x) / quot) 'x-Wert
    thePoint.y = Vertices(m).y + ((Vertices(m + 1).y - Vertices(m).y) / quot) 'x-Wert
    Exit For
  End If
Next m
Set AlongLine = thePoint 'Rückgabe
```

CODE 8: AUSZUG AUS AVENUE-SUBROUTINE ZUM AUFFINDEN VON ÜBERLAGERUNGEN

```
oldsel = theFtab.getSelection.clone 'Selektion der Attributtabelle speichern
Ueberlagerungen = {}'Ergebnislisten anlegen
Centers = {}
for each k in theFtab.getSelection 'Selektion der Attributtabelle durchlaufen
  aShape = theFtab.returnValue(theFtab.findField("Shape"), k) 'Abschnitt
  theFtab.selectByPolyline(aShape, #VTAB SELTYPE NEW) 'Tabelle mit Shape selektieren
  if(theFtab.getSelection.count < 2)then continue end '*Einzelllinie
  uebs = {} 'Ergebnisse pro Abschnitt
  for each rec in theFtab.getSelection  'aktuelle Selektion durchlaufen
    aLine = theFtab.returnValue(theFtab.findField("Shape"), rec) 'selektierter Abschnitt
```

```
    if(k.clone = rec.clone)then continue end'* 'wenn Datensätze übereinstimmen >> Abbruch
    int = aShape.lineIntersection(aLine)'Verschneidung
    if(int.returnLength = 0)then continue end 'wenn keine gemeinsame Strecke gefunden wurde ignorieren
    uebs.add(int)'zur Liste der aktuellen Überlagerungen hinzufügen
  end
  if(uebs.count = 0)then continue end 'keine Überlagerungen in Selektion gefunden
  for each u in uebs
    if(Centers.findByValue(u) <> (-1))then 'Test auf Vorhandensein in Überlagerungsliste
     Ueberlagerungen.add(u) 'zur Liste der Überlagerungen hinzufügen
     Centers.add(u.returnCenter)
    end
  end
  theFtab.setSelection(oldsel) 'Selektion wieder herstellen
end

return {Ueberlagerungen, Centers}
```

CODE 9: AUSZUG AUS AVENUE-SUBROUTINE ZU KORREKTUR DER ANORDNUNG ZUSAMMENZUFÜHRENDER LINIEN

```
theLines = self.get(0) 'Linien werden vom aufrufenden Script übergeben
startLine = self.get(1)'wird aus Subroutine ermittelt

'Ordnung der Linien
newLines = {StartLine}  'Array für neue Linien mit Startlinie initialisieren
while(theLines.count > 0) 'while-Schleife bis Abbruchbedingung erfüllt (true) ist
  for each b in (theLines.count-1)..0 'Durchlauf über die Linien (Startlinie wurde bereits entfern!)
    newLine = theLines.get(b)'aktuelle Linie an Variable übergeben
   'Tests auf Berührung von aktueller Linie und Startlinie >> falls nicht wird die aktuelle Linie übersprungen:
    if(newLine.intersects(StartLine).not)then continue end  ' >> nächste Linie in der Kette gefunden!!
    Ml1 = newLine.asMultipoint.asList 'Startlinie in ihre Vertices auflösen
    'Tests auf Berührung des Anfangs der aktuellem Linie mit  Startlinie
    if(Ml1.get(0).intersects(StartLine))then 'Linie copieren (clone) und umgedrehen (flip)
      newLine = theLines.get(b).clone.flip
    end
    newLines.add(newLine) 'aktuelle, ggf. kopierte und gedrehte Linie zum Ergebnis-Array hinzufügen
    startLine = newLine 'Linie als Startlinie festlegen
    theLines.remove(b) 'Linie aus Ausgangs-Array entfernen
    break 'Abbbruch der inneren Schleife
  end
end
return newLines 'Ergebnis-Array an aufrufendes Script übergeben
```

CODE 10: AUSSCHNITT AUS AVENUE-SUBROUTINE ZUR RICHTUNGSKORREKTUR UND ZUM AUFFINDEN VON RINGSTRUKTUREN

```
TheFtab = self.get(0) 'kommt von aufrufendem Script
oldsel = theFtab.getSelection.clone 'aktuelle Selektion des aktiven Themas speichern
theKey = av.run("RUN SelectByPoint", nil)'Datensatznummer des angeklickten Auslasses
'fs = Namen zu bearbeitender Felder
av.run("RUN InitializeFields", {{"Visited", "Net"},  0}) 'Inhalt der Felder auf 0 setzen
theFtab.SetValue(theFtab.findField("Visited"), theKey, 1)) 'Datensatz des Starts auf Visited setzen
startLine = av.run(("RUN ReturnStartline", {theKey, theFtab}) 'Start, ggf. drehen
sf = theFtab.findField("Shape") 'Shapefeld an Variable übergeben
vis = theFtab.findField("Visited") 'Feld „Visited“ an Variable übergeben
beginners = {StartLine} 'Array für gegenwärtiges Level mit Startlinie initialisieren

done = false 'Abbruchbedingung
while (done = false)
  retList=av.run("RUN_ReturnStarts", beginners, theFtab) ' nächstes Level ermitteln
        newBeginners = retList.get(0) 'nächstes Level in Variable schreiben
   foundRecurse = retList.get(1) 'Anfang von Kreisstruktur gefunden
   testRecurse = retList.get(2)
   newBeginners = av.run("RUN_ProcessLines", newBeginners) 'Linien vorbereiten
   if(foundRecurse and testRecurse) then 'Rücklauf
    if(fromShape <> nil)then 'Test, ob im Ring rechts oder links „gelaufen“ werden muss
     goRight = Start.intersects(b) 'Speicherung in boolscher Variablen
    end
    recurseShape = b 'Rücklaufvariable mit aktueller Linie des gegenwärtigen Levels belegen
    found = false 'Abbruchbedingung für While-Schleife festlegen
    While(found = false) 'innere While-Schleife anstossen
      ML = recurseShape.asMultipoint.asList 'Vertices der Linie in ML speichern
      Ende = Ml.get(Ml.count-1) 'Ende der Linie
```

```
      theFtab.selectbyShapes({Ende}, #VTAB SELTYPE new) 'Selektion der Themas mit Ende
      if(theFtab.getSelection.count = 1)then found=true continue end 'Quelle gefunden
      founds = {} 'Array für Nachbarn in Rücklaufrichtung initialisieren
      for each k in theFtab.getSelection 'Nachbarn untersuchen
        s = theFtab.returnvalue(sf, k) 'Nachbar an Variable übergeben
        if(s.returnCenter.intersects(recurseShape))then continue end 'ist Rücklauflinie
        thisStart = s.asMultipoint.asList.get(0) 'Anfang des Nachbarn
        if(thisStart.intersects(recurseShape).not)then 'Test auf Berührung mit akt. Rücklauflinie
          s = s.flip 'ggf. drehen
        end
        founds.add(s) 'zum Array der Nachbarn hinzufügen
      end 'Ende der Schleife über alle Nachbarn
      if(founds.count > 1)then 'Verzweigung gefunden
        'rechten oder linken Nachbarn finden (vgl. Absch. 3.2.5.2)
       founds=av.run("Run RechtsLinks",{recurseShape.flip, founds, goright})
      end 'durch den Rechts-Links-Tests wird hier der rechteste Nachbar genommen
      recurseShape=founds.get(0) 'Zuweisung an Variable
      ML = recurseShape.asMultipoint.asList 'Vertices des gefundenen nächsten Nachbarn
    end 'Ende der Rücklauf-While-Schleife
   end 'Ende der Bedingung für den Test
 end 'Ende für dieses Level
 if(newBeginners.count = 0)then done=true end '*Abbruchbedingung erfüllt >> Ende der äußeren Schleife
 Beginners = newBeginners 'Linien des nächsten Levels an Array übergeben
end 'Ende der äusseren Schleife
theFtab.EndTransaction 'Ende der Bearbeitung
theFtab.StopEditingWithRecovery(True) 'Speicherung der Bearbeitung
theFtab.setSelection(oldsel) 'Selektion zurücksetzen
av.PurgeObjects 'Garbage Collector leeren
```

CODE 11: AVENUE-SUBROUTINE ZUR ZUWEISUNG DER UNTER-OBERLIEGER-BEZIEHUNGEN

```
for each r in theFtab 'jeden Eintrag in der Attributtabelle durchlaufen
  aShape = theFtab.ReturnValue(theFtab.findfield("shape"), r) 'Linie einlesen
  ML = aShape.asMultipoint.asList 'Linie in Vetices zerlegen
  if(ML2.count < 2)then continue end 'korrupte Linien ausschliessen
  Ende = ML.get(ML.count-1) 'Endpunkt der Linie
  theFtab.SelectbyShapes({Ende},#VTAB SELTYPE New) 'alle Abschnitte mit Ende selektieren
  if(theFtab.GetSelection.Count=1)then continue end 'Gebietsauslass erreicht
  for each k in theFtab.getSelection 'alle selektierten Datensätze durchlaufen
   if(r=k)then continue end 'betrachteten Abschnitt ausschliessen
   theShape = theFtab.ReturnValue(theFtab.findfield("shape"),k)) 'UL einlesen
   FGWID = theFtab.returnValue(theFtab.findField("FGWID"), k)'FGWID
   theFtab.SetValue(theFtab.findField("FGWULID"), r, FGWID) 'FGWULID zuweisen
  end
end
```

CODE 12: VISUAL BASIC ROUTINE ZUR SELEKTION ALLER OBERLIEGER VON EINEM SELEKTIERTEN ABSCHNITT AUS

```
'FGWID des selektierten Abschnittes kommen von aufrufender Prozedur und wurden zuvor festgelegt
Function DisplaySubtree(FGWID)
 Dim Oberlieger as new Collection 'Collection
 'FGWIDS der Oberlieger erhalten
 FGWIDS = SQLConn ("SELECT FGWID, INFO  FROM river WHERE FGWULID=FGWID")
 If(FGWIDS = -1)Then 'Alle Oberlieger gefunden
   Set DisplaySubtree = Oberlieger 'Oberlieger als Rückgabewert
   End Function 'Funktion beenden und zurück zur aufrufenden Routine
 End
 For i = 0 to Ubound(FGWIDS) 'Iteration über alle Oberlieger
    DisplaySubtree (FGWIDS(i)) 'Rekursion
    Oberlieger.Add(FGWIDS(i)) 'Oberlieger auffüllen
 Next i
 Set DisplaySubtree = Oberlieger 'Oberlieger als Rückgabewert
End Function
```

CODE 13: STORED PROZEDURE FÜR STACK-ALGORITHMUS

```
CREATE PROC expand (@current char(20)) AS SET nocount on test 'Prozedur anlegen
DECLARE @level int, @line char(20) 'Variablen deklarieren
CREATE TABLE #stack (item char(20), level int) 'Tabellen erzeugen
INSERT INTO #stack VALUES (@current, 1)'Stack-Tabelle auffüllen
SELECT @level = 1 'Gebietsauslass
WHILE @level > 0 'alle Oberlieger:
 BEGIN 'jeweilige Kinder abfragen bis alle Oberlieger gefunden sind
  if EXISTS (SELECT * FROM #stack WHERE level = @level)
    BEGIN
      SELECT @current = item
      FROM #stack
      WHERE level = @level
      SELECT @line = space(@level - 1)  @current
      PRINT @line
      DELETE FROM #stack
      WHERE level = @level
        AND item = @current
      INSERT #stack
      SELECT child, @level  1
      FROM hierarchy
      WHERE parent = @current
      if @@rowcount > 0
        SELECT @level = @level  1
    END
  else
    SELECT @level = @level - 1
END
```

CODE 14: PSEUDOCODE ZUR DATENSELEKTION AUF BASIS ANGRENZENDER LISTEN IN EINER FLACHEN TABELLEN

```
Function DisplayChildren()
  SELECT FGWID, INFO, HIEDIST 'SQL-Statement
  FROM river
  ORDER BY POSITION
  while das Recordset nicht leer ist {
      (indent by Hiedist)
  }
End Function
```

CODE 15: AVENUE-CODE ZUR SELEKTION ALLER OBERLIEGER MITTELS INDIZIERTER ARRAYS

```
Starts = {StartFGWID}
QueryString = "([FGWID]="+""+StartFGWID.asString+""+") or " 'GIS-Abfrage initiieren
while (true) 'erzwungener Durchlauf
  newStarts = {}
  for each s in Starts
    ind = 0
    while(ind <> -1)
      ind = FGWULIDs.findByValue(s) 'Position in der Unterliegerliste ermitteln
      if(ind = -1)then break end 'Quelle gefunden
      FGWID = FGWIDs.get(ind) 'nächsten Eltern-Abschnitt einlesen
      QueryString = QueryString + "([FGWID] = "+""+ FGWID+""+") or "
      newStarts.add(FGWID) 'GIS-Abfrage aufbauen
      for each l in {FGWIDs, FGWULIDs, Points}
        l.remove(ind) 'Listen aus Performance-Gründen „aufräumen"
      end
    end
  end
  Starts = newStarts '„Eltern"-Liste aktualisieren
  if(Starts.count = 0)then break end'alle Oberlieger gefunden und Abbruch der Schleife
end
'Sql-basierte GIS-Abfrage abschließen
QueryString = QueryString.left(QueryString.count - (") or ").count))
QueryString = QueryString + "))"
'GIS-Abfrage absenden
theFtab.query(QueryString, theFtab.getSelection, #VTAB_SELTYPE_NEW)
```

CODE 16: AVENUE-CODE ZUR SELEKTION ALLER UNTERLIEGER MITTELS INDIZIERTER ARRAYS

```
founds = {}
QueryString = "" 'GIS-Abfrage initiieren
for each f in 0..(StartFGWULIDS.count-1) 'Unterlieger der selektierten Abschnitte
  StartFGWULID = StartFGWULIDs.get(f) 'erster Unterlieger
  if(StartFGWULID < 1)then continue end  'Gebietsauslass gefunden, ggf. nächster Startabschnitt
  Start = StartFGWULID
  while (2<>1) 'erzwungener Durchlauf
    ind = FGWIDs.findByValue(Start)
    if(ind = -1)then break end'*
    newStart = FGWULIDs.get(ind)
    FGWID = FGWIDs.get(ind)
    QueryString="([FGWID]="+""+FGWID.asString+""+")or " 'GIS-Abfrage erweitern
    for each l in {FGWIDs, FGWULIDs, Points}
      l.remove(ind) 'Listen aus Performance-Gründen „aufräumen“
    end
    Start = newStart ‘„Kind“ aktualisieren
    if(Start < 1)then break end 'Gebietesauslass gefunden
  end
end
'Sql-basierte GIS-Abfrage abschließen
QueryString = QueryString.left(QueryString.count - (") or ").count))
QueryString = QueryString + "))"
'GIS-Abfrage absenden
theFtab.query(QueryString, theFtab.getSelection, #VTAB_SELTYPE_NEW)
```

CODE 17: SQL-CODE ZUR ERZEUGUNG VON TAB. 10

```
CREATE TABLE Welt(
Parent VARCHAR2(30) REFERENCES Welt,
Name  VARCHAR2(30) PRIMARY KEY);

INSERT INTO Welt VALUES (NULL, 'Welt') ;
INSERT INTO Welt VALUES ('Welt', 'Europa') ;
INSERT INTO Welt VALUES ('Europa', 'England') ;
INSERT INTO Welt VALUES ('Europa', 'Niederlande');
INSERT INTO Welt VALUES ('Europa', 'Deutschland') ;
INSERT INTO Welt VALUES ('Welt', 'Asien');
INSERT INTO Welt VALUES ('Asien', 'Japan');
INSERT INTO Welt VALUES ('Asien', 'China');
INSERT INTO Welt VALUES ('Welt', 'Amerika');
INSERT INTO Welt VALUES ('Amerika', 'United States');
INSERT INTO Welt VALUES ('Amerika', 'Mexico');
INSERT INTO Welt VALUES ('Welt', 'Afrika');
INSERT INTO Welt VALUES ('Afrika', 'Ägypten');
INSERT INTO Welt VALUES ('Afrika', 'Morokko');
```

CODE 18: AUSSCHNITT AUS AVENUE-IMPLEMENTIERUNG DER STRAHLER-ERFASSUNG NACH LANFEARE

```
'erzeugt ein Dictionary von ToNodes-Ids und den Datensatznummern der jeweils in diesen Node fließenden Abschnitte. Durch die Aufnahme der Re-
cord'nummer kann später die Bitmap der Attributtabelle direkt bearbeitet werden anstatt sie mit einer Query auf die gesamte Tabelle updaten zu müssen
BM = theFTab.GetSelection
BM.ClearAll
BMSize = BM.GetSize
ToNodeDictionary = Dictionary.Make(1)

For Each rec In 0.. (BMSize - 1)
   ToNodeID = theFtab.ReturnValue(theToNodeField,rec)
   RecList = ToNodeDictionary.Get(ToNodeID)
   If (RecList = NIL) Then 'Neue ToNode-ID zum Dictionary hinzufügen
       ToNodeDictionary.Add(ToNodeID,{rec})
       ToNodeDictionary.SetSize((ToNodeDictionary.GetSize + 1))
   Else
       RecList.Add(rec) 'ToNode-ID gefunden; Recordnummer in die Liste der Oberlieger schreiben
       ToNodeDictionary.Set(ToNodeID, RecList) ‘und diese zum Dictionary hinzufügen
   End
End

'alle Strahlerordnungen aufsummieren. Diese wurde zuvor fakultativ auf 1 initialisiert
strahler_sum = 0
For Each rec In theFtab
   strahler sum = strahler sum + theFtab.ReturnValue(theStrahlerField,rec)
End
```

Codes

```
'Hauptschleife des Ordnungsprozesses
goOn = TRUE
recheck = 0
goneThrough = 0
While (goOn)
    For Each Main rec In theFtab
       FromNode  = theFtab.ReturnValue(FromNodeField, Main rec)
       Upstreams = ToNodeDictionary.Get(FromNode)
       If (Upstreams = NIL) Then 'Quelle gefunden >> ignorieren
       Else 'Maximale Strahlerordnung der direkten Oberlieger ermitteln und deren (gespeicherte) Datensatznummer  selektieren
            BM.ClearAll
            For Each upstream_arc In Upstreams
               BM.Set(upstream_arc)
            End
            theFtab.UpdateSelection
            Smax = 0
            For Each rec In BM
               y = theFtab.ReturnValue(theStrahlerField,rec)
               If (y > Smax) then
                  Smax = y
               End
            End
            'alle Datensätze, die Strahler kleiner als Smax haben aus der Selektion entfernen
            CopyBM = BITMAP.Make(BMSize) 'Neue Bitmap anlegen
            For Each k In BM
               Strahler = theFtab.ReturnValue(theStrahlerField,selected_record)
               If (Strahler = Smax) Then
                  CopyBM.Set(selected record)   'Neue Bitmap mit Recordnummer des Abschnittes besetzen
               End
            End
            theFtab.SetSelection(CopyBM)
            BM = theFtab.GetSelection

            'Test, ob die FromNode-ID der direkten Oberlieger dieselben Nodes haben
            FromOrigList = List.make 'Liste anlegen
            For Each rec In BM
               FromOrigList.Add(theFtab.ReturnValue(theFromOrigField,rec))
            End
            FromOrig_1 = FromOrigList.Get(0) 'ersten der Oberlieger in Variable schreiben
            found = TRUE
            For Each x In FromOrigList
               If (FromOrig_1 <> x) Then found = FALSE  End'*
            End
            If (found) Then 'Strahler erhöht sich nicht
                theFtab.SetValue(theStrahlerField,Main rec, Smax)
                theFtab.SetValue(theFromOrigField,Main rec, FromOrig 1)
            Else 'Strahler erhöht sich
                theFtab.SetValue(theStrahlerField,Main_rec, (Smax + 1))
            End
       End
    End 'Main_rec
    recheck = recheck + 1
    count = 0
    For Each rec In theFtab 'Testlauf über alle Abschnitte, ob sich Strahler verändert hat
       count = count + theFtab.ReturnValue(theStrahlerField, rec)
    End
    If (count <> strahler_sum) then
        goOn = TRUE
        strahler sum = count
        goneThrough = 0
    Else
        goneThrough = goneThrough + 1
    End
    If (goneThrough = 3) Then 'Sicherheitsbedingung für Abbruch in bei verschachtelten Ringstrukturen
       goOn = FALSE
    End
End 'While
```

Codes

CODE 19 JAVA-FUNKTION ZUR ERMITTLUNG DES KLEINSTEN STARTWERTES FÜR DEN WEG-SCHLÜSSEL BEI DER AUFWÄRTSGERICHTETEN ABFRAGE

```
Public String findeWEG(String WEG){ //Deklaration der Funktion mit Parameterübergabe (Start-Code)
int k=0; //Initalisierung der Variablen
String Rueckgabe="", xx = "";
for (int x=(WEG.length()-1); x > 0; x--){ //Zeichenkette von rechts nach links
          if(k!=0) continue; //Null ignorieren
          xx = WEG.substring(x, x-1 ); //Ziffer zurückgeben
          if(xx != "0"){ //Erste gerade Zahl finden und alle bis dorthin gefundenen Zahlen durch Null ersetzen
            if(Integer.valueOf(xx) mod 2 != 0) Rueckgabe = "0" + Rueckgabe;
            else Rueckgabe = xx + Rueckgabe;
            k = x;
         else  Rueckgabe = "0" + Rueckgabe;
       }
       Rueckgabe = WEG. substring (0, k)+ Rueckgabe; //Rückgabestring
       return Rueckgabe; // Verlassen des Algorithmus und Rückgabe an das die aufrufende Funktion / Klasse
}
```

CODE 20 JAVA-FUNKTION ZUM AUFFINDEN DER UNTERLIEGER EINES BESTIMMTEN WEG-SCHLÜSSELS

```
Public String[] findWEGDown(String WEG){ //Deklaration der Funktion mit Parameterübergabe (Start-Code)
   public String selectDownStream(String Auslauf){//Variablen-Deklaration
   String WEG = theTable.returnValueString(wegField, theKey);
   String WEGs[] = String[theTable.getNumRecords()];
   String astr = "";
   Int a = 0;
   theTable.MoveFirst();//Record zurücksetzen
   while(Not(theTable.EOF)){ //Durchlauf der Tabelle
     String w= theTable.Value(wegField);
     if(WEGs.find(w)<>-1)continue;//Übergehen, wenn Schlüssel bereits in der Liste ist
     if((Double)w<(Double)WEG){continue};//Casting des Schlüssels und Vergleich mit  WEG, ggf. übergehen
     Take = compareParts(WEG, w); //Übergabe an Funktion zur String-Analyse (siehe CODE 10.3.1.2)
     if(Take) WEGs[a] = w; //zum Array der WEGs hinzufügen
     theTable.MoveNext();
   }
   return Sort(WEGs, True); //Rückgabe der aufsteigend sortierten WEG-Schlüssel
}
```

CODE 21: JAVA-FUNKTION ZUM TEST EINES WEG-SCHLÜSSELS AUF UNTERLIEGEREIGENSCHAFT

```
Public Boolean compareParts (String WEG, String Aktuell){ //Deklaration und Parameterübergabe
   Boolean Test = true;
   String m;
   for (int x = (Aktuell.length()-1); x > 0; x--){//sukzessive von rechts...
     m = Aktuell.substring(x, x-1); //Nullen ignorieren
     if(m = "0") continue;  //Frage 1
     if(Integer.valueOf(m) mod 2 != 0){ //Fragen 2 und 3
         if(WEG.left(x+1) <> Aktuell.left(x+1)){
           Test = FALSE; //ist unerwünscht >> Abbruch der Schleife
           Break;
         }
     }
   }
   return Test; //Rückgabe des boolschen Wertes
}
```

CODE 22: AUSSCHNITT AUS AVENUE-ROUTINE ZUR ERFASSUNG UND SPIEGELUNG VERSCHACHTELTER HIERARCHIEN

```
while (true)
    theFtab.selectByShapes({Start}, #VTAB_SELTYPE_NEW)
    founds = {}
    for each k in theFtab.getSelection
      s = theFtab.returnValue(TheFtab.findField("Shape"), k)
      if(s.returnCenter.Intersects(Startshape))then continue end'*
      if(theFtab.returnValue(TheFtab.findField("Done"), k) = 1)then continue end'*
      ML = Startshape.asMultipoint.asList
      Ende = ML.get(ML.count-1)
      founds.add(s)
    end

    if(founds.count = 0)then 'ggf. Zurücklaufen
```

```
      while (founds.count = 0)
        theFtab.selectByShapes({Ende}, #VTAB SELTYPE NEW)
        if(theFtab.getSelection.count < 2)then break end'*
        for each k in theFtab.getSelection
          if(theFtab.returnValue(TheFtab.findField("Done"), k) = 1)then continue end'*
          Startshape = theFtab.returnValue(TheFtab.findField("Shape"), k)
          founds.add(Startshape)
        end
        if(founds.count = 0)then
          for each k in theFtab.getSelection
              if(theFtab.returnValue(TheFtab.findField("rechts"), k) <> 0)then continue end'*
              theFtab.SetValue(theFtab.findField("rechts"), k, leftcount)
              leftcount = leftcount+1
              Startshape = theFtab.returnValue(TheFtab.findField("Shape"), k)
              ML = Startshape.asMultipoint.asList
              Ende = ML.get(ML.count-1)
              break
          end
        else break end'*
      end
    end
    founds = av.run("rivertool_Run_checkFounds", {Startshape, founds})'Ergebnisse überprüfen
    if(founds.count = 0)then break end'*

    'Oberlieger in Uhrzeigersinn ordnen
    if(founds.count > 1)then
       founds = av.run("rivertool Run RechtsLinks", {Startshape, founds, true})
    end
    theFtab.selectByShapes({founds.get(0).returnCenter}, #VTAB_SELTYPE_NEW)

    'Verschachtelungen in Tabelle spiegeln
    for each k in theFtab.getSelection
      s = theFtab.returnValue(TheFtab.findField("Shape"), k)
      if(theFtab.returnValue(TheFtab.findField("Done"), k) = 1)then continue end'*
      theFtab.SetValue(theFtab.findField("Done"), k, 1)
      theFtab.SetValue(theFtab.findField("links"), k, leftcount)
      leftcount = leftcount+1
    end
    Startshape = leftest

    ML = Startshape.asMultipoint.asList
    if(Ml.count < 2)then continue end'*
    Start = Ml.get(0)
    Ende = Ml.get(Ml.count-1)
    theFtab.selectByShapes({Start}, #VTAB_SELTYPE_NEW)
    if(theFtab.getSelection.count < 2)then
      theFtab.selectByShapes({Startshape.returnCenter}, #VTAB SELTYPE NEW)
      for each k in theFtab.getSelection
        theFtab.SetValue(theFtab.findField("rechts"), k, leftcount)
      end
      leftcount = leftcount+1
    end
end
```

CODE 23: PSEUDOCODE ZUM DIJKSTRA-ALGORITHMUS

/*Der Dijkstra-Algorithmus berechnet bei Eingabe eines Startknotens s alle kürzesten Wege in einem zusammenhängenden, gerichteten Graphen, bei dem die Kantengewichtungen nicht-negativ sind. Die einzelnen Knoten des zusammenhängenden gerichteten Graphen sind als Integer repräsentiert. Der Integerwert s entspricht damit dem Startknoten.*/

```
Algorithmus Dijkstra (int s);
// Initialisierung Start
IncludedList = {};
CandidatesList = {s};
DistList[s] = 0;
for (each w aus (V-{s})) DistList[w] = oo;
    // Initialisierung Ende
    // Berechnungsschritt Start
    while (CandidatesList != (leere Menge)){
          // Der kürzeste Weg zum Knoten s ist bereits für alle Knoten in IncludedList und mindestens einen Knoten x aus CandidatesList berechnet.
          wähle den Knoten w aus CandidatesList mit minimalem Wert DistList[w];
          IncludedList = IncludedList + {w};
          CandidatesList = CandidatesList - {w};
          for (each (w,l) aus E) { // Zugriff auf Adjazenzmatrix
            if ((l kein Element aus IncludedList) && (DistList[w] + cost(w,l) < DistList[l])){
              DistList[l] = DistList[w] + cost(w,l);   // (*)
```

```
                CandidatesList = CandidatesList + {l};
            }
          }
    }
}
```

/* Wenn ein Knoten z in die IncludedList aufgenommen wird, so ist gewährleistet, dass dist[z] (Weglänge vom Startknoten s zum Knoten z) minimal ist, also die kürzeste Weglänge enthält. Da sukzessive alle Knoten des Graphen in die Menge IncludedList aufgenommen werden, ist nach Beendigung des Algorithmus sicher gestellt, dass die kürzesten Wege zu allen Knoten des Graphen in DistList [] vorhanden sind.*/

CODE 24: PHP-FUNKTION ZUR DARSTELLUNG DER RÜCKGABE EINER DATENBANKABFRAGE

```
function print result table($MainQuery, $fields){
  $result = connectToMySql($MainQuery);
  echo "<table TABLE BORDER = 0 Cellpading = 0 Width=100% BGCOLOR=rgb(132,176,225)><TR
ALIGN=\"LEFT\">";
  //echo "<th BGCOLOR=9999cc><FONT SIZE=1>query</th>\n";
  for ($i = 0; $i < count($fields); $i++){
      echo "<th BGCOLOR=rgb(52,125,207)><FONT SIZE=1 Face=\"Arial\" COLOR=WHITE>
      ".$fields[$i]."</th>\n";
  }
  echo "</tr>\n";

 // Alle Ergebniszeilen durchgehen
  while ($row = mysql_fetch_row($result)){
    echo "<tr ALIGN=\"LEFT\">\n";
    //echo "<td><IMG SRC=\"/pic/examine.gif\"></td>";
    for ($i = 0; $i < mysql num fields($result); $i++){
       $fn = mysql field name($result,$i);
       if(in array($fn, $fields)){
          echo "<td><FONT SIZE=1 FACE = \"ARIAL\">$row[$i]</FONT></td>";
       }
    }

    echo "</tr>";
  }
  echo "</table>";
  mysql_free_result ($result);
  return $retlist;
}
```

CODE 25: PHP-FUNKTION ZUR ERMITTLUNG EINES ZU SELEKTIERENDEN RAUMAUSSCHNITTES

```
function getExtent($MainQuery){
  $result = connectToMySql($MainQuery);
  $minx = 0;
  $miny = 0;
  $maxx = 0;
  $maxy = 0;
  while ($row = mysql fetch row($result)){
    for ($i = 0; $i < mysql_num_fields($result); $i++){
       $fn = mysql field name($result,$i);
       $type = mysql field type($result,$i);
       //Extent auslesen
       if(strtoupper($fn) == "X"){
          if($minx == 0)$minx=$row[$i];
          if($minx > $row[$i])$minx=$row[$i];
          if($maxx == 0)$maxx=$row[$i];
          if($maxx < $row[$i])$maxx=$row[$i];
       }
       if(strtoupper($fn) == "Y"){
          if($miny == 0)$miny=$row[$i];
          if($miny > $row[$i])$miny=$row[$i];
          if($maxy == 0)$maxy=$row[$i];
          if($maxy < $row[$i])$maxy=$row[$i];
       }
    }
  }
  $numberFields = array();
  $Fields = array();
  for ($i = 0; $i < mysql num fields($result); $i++){
       $fn = mysql_field_name($result,$i);
       $type = mysql_field_type($result,$i);
       if($type == real) $numberFields[] = $fn;
       $Fields[] = $fn;
  }
```

```
  //mysql_free_result ($result);
  $extent = array($minx, $maxy, $maxx, $miny);
  $retlist = array($extent, $Fields, $numberFields);
  return $retlist;
}
```

CODE 26: PHP-FUNKTION ZUR ABWÄRTS GERICHTETEN ABFRAGE

```
function Downstream($table, $sidecode, $fields, $FldsSel){
   //Return Sidecodes
    $str = "('".$sidecode."'";
    for($i=strlen($sidecode); $i>=0; $i--){
       $str .= ", '".substr($sidecode,0,  $i)."'";
    }
     $str.= ")";
    $query = "SELECT * FROM $table WHERE SIDECODE in $str ORDER BY SIDECODE DESC";
    print result table($query, $fields);
    echo "<HR>";
    $query = "SELECT Hiname, $FldsSel FROM $table WHERE SIDECODE in $str ORDER BY SIDECODE
    DESC";
    $result = connectToMySql($query);
    makeChart($result, $FldsSel);
    return $query;
}
```

CODE 27: PHP-FUNKTION ZUR AUFWÄRTS GERICHTETEN ABFRAGE

```
function print result table($MainQuery, $fields){
  $result = connectToMySql($MainQuery);
  echo "<table TABLE BORDER = 0 Cellpading = 0 Width=100% BGCOLOR=rgb(132,176,225)><TR
  ALIGN=\"LEFT\">";
  //echo "<th BGCOLOR=9999cc><FONT SIZE=1>query</th>\n";
  for ($i = 0; $i < count($fields); $i++){
      echo "<th BGCOLOR=rgb(52,125,207)><FONT SIZE=1 Face=\"Arial\" COLOR=WHITE>
      ".$fields[$i]."</th>\n";
  }
  echo "</tr>\n";
 // Alle Ergebniszeilen durchlaufen
  while ($row = mysql fetch row($result)){
    echo "<tr ALIGN=\"LEFT\">\n";
    //echo "<td><IMG SRC=\"/pic/examine.gif\"></td>";
    for ($i = 0; $i < mysql_num_fields($result); $i++){
       $fn = mysql field name($result,$i);
       if(in array($fn, $fields)){
          echo "<td><FONT SIZE=1 FACE = \"ARIAL\">$row[$i]</FONT></td>";
       }
    }
    echo "</tr>";
  }
  echo "</table>";
  mysql free result ($result);
  return $retlist;
}
```

CODE 28: PHP-FUNKTION ZUR DARSTELLUNG EINES TABELLEN-TREEVIEWS

```
function TableTreeview($table, $sidecode, $FldsSel, $Radio){
  //SIDECODES
  $query = "SELECT SIDECODE FROM $table WHERE SIDECODE = '$sidecode'";
  $result = connectToMySql($query);
  $StartSidecode = returnSidecode($result);
  mysql free result ($result);
  $Startlength = strlen($StartSidecode);

  $query = "SELECT * FROM $table WHERE SIDECODE like '$sidecode%' ORDER BY SIDECODE";
  $result = connectToMySql($query);
  echo "<TABLE BORDER=0 Cellpading=0 BGCOLOR=rgb(132,176,225)><tr>";
  while ($row = mysql fetch row($result)){
    echo "<tr>";
    $Infostr = "";
    $HINAME = "";
    $sidecode = "";
    $Hipoints = "";
    for ($i = 0; $i < mysql_num_fields($result); $i++){
```

```
        $fn = mysql_field_name($result,$i);
        $Infostr .= "$fn=$row[$i],  ";
        if(strtoupper($fn) == "SIDECODE")
            $sidecode = $row[$i];
        if(strtoupper($fn) == "HINAME")
            $HINAME = $row[$i];
        if(strtoupper($fn) == strtoupper("Hipoints"))
            $Hipoints = $row[$i];
    }
    $Description = "\"images\Verzweigung.jpg\"";
    if($Hipoints == 1) $Description="\"images\Quelle.jpg\"";
    $diff = strlen($sidecode)-$Startlength+1;
    //echo "<TD> <IMG SRC=\"images/examine.jpg\"></TD>";
    echo "<TD COLSPAN=$diff bgcolor=rgb(52,125,207)><A
    href=formular.php?SelectName=$HINAME&FieldsSelect=$FldsSel&Radio=$Radio onMouse
    Over=\"alert('$Infostr')\">".
    "<IMG SRC=$Description></A></TD>";
    for ($j=1; $j<=$diff; $j++){
        //echo "<td bgcolor=blue>.</td>";
    }
    $inserString ="<A
    href=formular.php?SelectName=$HINAME&FieldsSelect=$FldsSel&Radio=$Radio>
    $HINAME</A>";
    echo"<td><FONT SIZE=1 COLOR=rgb(52,125,207) Face=Arial>$inserString</FONT><TD></tr>";
  }
  echo "</TABLE>";
  mysql free result ($result);
  return $query;
}
```

CODE 29: PHP-CODE ZUR DARSTELLUNG EINES RECURSIVEN-TREEVIEWS

```
<?php
function  eingangAnzeigen($href="",$Ziel,$iconindex="0",$text="Fehler:  Sie  haben  keinen  Text
uebergeben!",$schluessel,$zeilenweite){
  global $icons, $cssKlassen, $Zaehler, $tabellenWeite;
  $Zaehler++;
  if ($iconindex < count($icons)){
     if ( checkGeoeffnet($schluessel) == "" ){
       $iconindex++; // Ordner geöffnet darstellen
       $icon = "<img src=\"".$icons[$iconindex]."\" align=\"left\" border=0 vspace=1>";
     }else
       $icon = "(Iconindex $iconindex wurde nicht definiert.)";
     if (!empty($Ziel)) $t = " target=\"$Ziel\"";
     return "<td class=\"".$cssKlassen[$iconindex]."\" width=\"".($tabellenWeite -
     $zeilenweite)."\"><a href=\"$href\"$t>$icon$text</a></td>";
  }
}

function checkGeoeffnet($which){
  global $lfdnr;
  for ($w=0;$w<count($lfdnr);$w++){
    if ($lfdnr[$w] == $which) $ok = 1;
  }
  if ($ok != 1) return "&lfdnr[]=$which";
}

function rekursiv($zu,$eingang, $spaltenAnzahl){
  global $lfdnr, $PATH_INFO, $platzhalter, $tabellenWeite;
  $zeilenweite = $spaltenAnzahl * $platzhalter + 1; // Netscape!!
  $spaltenAnzahl++;
  for (reset($eingang);current($eingang);next($eingang)){
      $e = current($eingang);  // wenn Zuordnung gleich x (übergeordneter Verzeichnisse)
      if ($e[0] == $zu){// Icon-Index und Link? (0=folder, 1=openfolder, >1=doc)
         if ($e[4] < 2){
             unset($durchlauf);
             $durchlauf = "?sid=1";
             for ($w=0;$w<count($lfdnr);$w++)
               if ($lfdnr[$w] != key($eingang))
                 $durchlauf .= "&lfdnr[]=".$lfdnr[$w];
                 $link = $PATH_INFO.$durchlauf.checkGeoeffnet(key($eingang));
               }elseif ($e[4] > 1)  $link = $e[2];
               // HTML-Ausgabe
               if ($e[0] == 0 and is_int(key($e))) $bgcolor = "#EEEEEE";
               print "<table cellpadding=0 cellspacing=0 border=0 width=\"$tabellenWeite\"><tr
               bgcolor=\"$bgcolor\"><td><img src=\"spacer.gif\" width=$zeilenweite
```

Codes

```
                height=5></td>";
                print eingangAnzeigen($link,$e[3],$e[4],$e[1],key($eingang),$zeilenweite);
                print "</tr></table>\n";

                // soll dieser Ordner geöffnet werden?
                unset($ok);
                $ok = 1;
                for ($w=0;$w<count($lfdnr);$w++){
                  if ($lfdnr[$w] == key($eingang)){
                     $ok = 1;
                     break;
                  }
                }
                if ($ok == 1) rekursiv(key($eingang),$eingang,$spaltenAnzahl);
              }
            }
          }
    }
}
```

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbständig und ohne fremde Hilfe verfasst, andere als die von mir angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzten Werken wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Thomas Koschitzki, *Halle, am* 12.01.2004

Angaben zur Person und zum Bildungsgang

Name:	Thomas Koschitzki
Geburtsdatum:	29.11.1966
Geburtsort:	Roßlau (Elbe)
Familienstand:	ledig

Theorie

1983 bis 1985	Erweiterte Oberschule (EOS) Roßlau; Abitur
10/1991 bis 09/1993	Studium der Architektur an der Hochschule Anhalt in Dessau
10/1993 bis 12/1999	Studium der Geographie an der Martin-Luther-Universität Halle/Wittenberg, Nebenfächer: Geologie, Botanik, Soziologie; Diplom
seit 04/2000	Dissertation zum Thema *"GIS-basierte, automatische Erfassung natürlicher Fliessgewässerhierarchien und ihre Abbildung in Datenbanken, beispielhaft dargestellt am Einzugsgebiet der Salza"* (Graduiertenstipendium des Landes Sachsen-Anhalt)

Praxis

10/1995 bis 03/1996	Auslandspraktikum in Mombasa/Kenya: - Geländekartierung des BAMBURI NATURAL PARK der dortigen Portland Zement Company - Erstellung eines touristischen Informationskonzeptes
02/1997 bis 04/1997	Auslandspraktikum am Remote Sensing Laboratory des Jacob Blaustein Institute For Desert Research in Sde Boker/Israel - Geländekartierung - Satelliten-gestützte Auswertung der Vegetationsbedeckung in der Negev-Wüste
02/1999 bis 04/1999	Studienaufenthalt am Computer Centre for Water Research (CCWR) / University of Natal (Südafrika) – Erarbeitung eines Fliessgewässerinformationssystems