

GeoCafé: Kommunikationszentriertes Gruppenlernen von Methoden der raumbezogenen Datenverarbeitung

Thomas BODE, Ingo DEVOOGHT, Thomas H. KOLBE,
Jörg STEINRÜCKEN und Markus WON

1 Einleitung

Einer der drei Bausteine des Vorhabens „Geoinformation – Neue Medien für ein neues Querschnittsfach“ ist neben der Erstellung generischer Lernmodule und der Entwicklung einer Projektumgebung die Konzeption und Implementierung einer interaktiven und kooperativen Lernumgebung für raumbezogene Verfahren und Analysemethoden, das GeoCafé.

Die Vermittlung von Inhalten aus dem Bereich der Informatik ist fester Bestandteil der Ausbildung im interdisziplinären Studienfach Geoinformation. An erster Stelle sei hier das Verstehen und Erlernen von Verfahren und Analysemethoden zur effizienten Lösung raumbezogener Aufgabenstellungen genannt. Das grundlegende Verständnis solcher Verfahren ist entscheidende Voraussetzung für den effektiven Einsatz von Geoinformationssystemen; diese stellen sich den Studierenden damit nicht mehr als Black Box, sondern als ein Zusammenspiel von Komponenten, die sich aus elementaren Funktionen zusammensetzen, dar. Allerdings erfordert das Verstehen und Erlernen solcher Inhalte ein hohes Maß an mathematisch-informatischer Abstraktion von Sachverhalten und Prozessen. Während dies Informatik-Studierende erfahrungsgemäß als ein Kernthema ihres Studiums begreifen, spielt dieser Aspekt für Studierende der Geoinformation aufgrund der thematischen Breite des Faches eine deutlich kleinere Rolle. Verfahren und Prozesse erschließen sich diesen Lernenden häufig nur schwer, was u.a. an der oft instruktivistischen Vermittlung durch formale und häufig rein textbasierte Darstellungen (z. B. in Pseudo-Code) liegt.

Die konkrete Umsetzung in Algorithmen und zugehörige Datenstrukturen erfordert zusätzlich Kenntnisse einer modernen Programmiersprache. Seit wenigen Jahren werden hier meist objektorientierte Sprachen, beispielsweise Java, eingesetzt. Die sinnvolle und strukturierte Anwendung einer solchen Sprache ist allerdings nur dann möglich, falls über die grundlegende Syntax hinaus die Konzepte und Begrifflichkeiten der Objektorientierung, einschließlich deren Darstellung in UML (Unified Modeling Language, UML 2004), beherrscht werden. Diese Kette von Abhängigkeiten stellt insbesondere für Studienanfänger oftmals ein großes Problem dar – zumal für ein Element des Lernens, die praktische Programmierübung, meist nur professionelle und damit sehr komplexe Entwicklungsumgebungen wie Sun ONE Studio (SUN 2004) oder netBeans (NETBEANS 2004) zur Verfügung stehen und für einen weiteren „Overhead“ sorgen.

Die komponentenbasierte Anwendung GeoCafé greift diese Problematik auf und integriert verschiedene Lösungsansätze zum konstruktivistischen Erlernen von Verfahren und Prozessen aus dem Bereich der Geoinformation. Es unterstützt kooperatives Lernen durch Kommunikationselemente wie Chat und Whiteboard. Gegenstand der Kommunikation sind Algorithmen und Datenstrukturen, die unter einer vereinfachten Programmierumgebung mit dem Java-Dialekt GeoJava entwickelt und visualisiert werden können.

2 Konzepte zur Vermittlung informatischer Inhalte

Bei der Vermittlung von Algorithmen und Programmiersprachen, insbesondere in Büchern und in Lehrveranstaltungen an Hochschulen, liegt der Schwerpunkt meist entweder auf dem Verständnis der Algorithmen oder dem Erlernen der Programmiersprache:

- Algorithmen und Datenstrukturen werden durch beschreibenden Text, unterstützt von Grafiken (Buch) oder vorgefertigten Animationen (Lehrveranstaltung), vermittelt. Der Bezug zur konkreten Implementierung ist oft nur über Pseudo-Code gegeben (z. B. OTTMANN & WIDMAYER 1996). Falls die Umsetzung in einer konkreten Programmiersprache erfolgt, werden Grundkenntnisse vorausgesetzt oder kurz abgehandelt, eine strukturierte Einführung erfolgt meist nicht.
- Das Erlernen einer modernen objektorientierten Programmiersprache bewegt sich in einem permanenten Spannungsfeld aus Vermittlung grundlegender Syntaxelemente (Datentypen, Schleifen, Bedingungen) und für praktische Übungen bereits notwendigen Elementen der Objektorientierung. So ist beispielsweise in Büchern und Lehrveranstaltungen zu Java der Satz „Das müssen Sie jetzt noch nicht verstehen!“ bei der Umsetzung des „Hello World“-Programms innerhalb der dafür benötigten Funktion „public static void main(String args[])“ nicht unüblich. Die eigentlichen problembezogenen Algorithmen werden, sofern überhaupt, nur als einfache Beispiele gebraucht, ein tieferes Verständnis von deren Ablauf ist nicht von vorrangigem Interesse.

Statt dieses beschriebenen „entweder – oder“ erlaubt die Nutzung moderner Multimedia- und Kommunikationstechniken ein Vorgehen, das von einem „sowohl – als auch“ ausgeht und parallel in Programmierung und Algorithmen einführt. Dieses Vorgehen scheint besonders geeignet, die formal-abstrakten Inhalte der Informatik nach modernen pädagogisch-didaktischen Zielsetzungen, – der Vermittlung von anwendbarem Wissen sowie der Aneignung von Kooperations-, Selbststeuerungs- und Medienkompetenz (vgl. KOPP, ZABEL & MANDL 2001) – zu erlernen.

Ansatz zum Erreichen der genannten pädagogisch-didaktischen Ziele ist das konstruktivistische Lernen, d. h. es werden die aktiven Konstruktionsprozesse des Lernenden in den Vordergrund gestellt (KOPP, ZABEL & MANDL 2001). Fünf Prozessmerkmale sind für das Lernen nach diesem Ansatz von Bedeutung (REINMANN-ROTHMEIER & MANDL, 2001):

- Lernen als aktiver Prozess: Aktive Beteiligung ermöglicht dem Lernenden effektives Lernen; Voraussetzung sind Motivation und Interesse.
- Lernen als selbstgesteuerter Prozess: Kontrolle des eigenen Lernprozesses durch den Lernenden.
- Lernen als konstruktiver Prozess: Erwerb und Nutzung von Wissen ist nur dann möglich, falls es in vorhandenen Wissensstrukturen eingebaut und auf Basis individueller Erfahrungen interpretiert werden kann.
- Lernen als situativer Prozess: Wissen weist immer situative und kontextuelle Bezüge auf, der Erwerb von Wissen ist immer an einen spezifischen Kontext gebunden.
- Lernen als sozialer Prozess: Erwerb von Wissen durch Interaktion mit anderen

Diese Prozesse können im Hinblick auf das Erlernen von Algorithmen und Programmiersprache durch eine interaktive und kooperative Lernumgebung, die das praxisnahe explorative Entwickeln von Algorithmen unterstützt, besonders gut angestoßen werden.

Im folgenden sollen vier charakteristische Merkmale einer solchen Lernumgebung näher betrachtet werden. Neben „Kommunikation“ und „Interaktion“ sind dies die „Algorithmenanimation“ sowie die „Entzerrung der Komplexität“ im Vergleich zu professionellen Softwareentwicklungssystemen. Während die Algorithmenanimation den explorativen Charakter der Lernumgebung unterstützt, sichert die Entzerrung der Komplexität die einfache Möglichkeit der praktischen Einübung von Programmen und Algorithmen.

2.1 Kommunikationssysteme auf textueller Basis – Chat-Systeme

In den letzten Jahren hat sich die Art der Kommunikation stark verändert. Aufgrund dieser Entwicklung sind neue Arten des Lernens möglich geworden, sie orientieren sich am Konzept der virtuellen Gemeinschaften. Beim gemeinsamen Lernen in Gruppen dient der Austausch, der häufig auf Basis asynchroner Kommunikationsmedien wie Email oder Newsgroups betrieben wird, dazu, Probleme zu erörtern oder gemeinsame Arbeiten abzusprechen. Auch Chat-Systeme haben mittlerweile eine große Verbreitung erreicht. Allerdings werden sie vor allem im Bereich der Unterhaltung eingesetzt und dienen bisher nur selten dem ernsthaften Informationsaustausch.

Gerade aber virtuelle Gemeinschaften, deren Ziel ein ernsthafter Informations- oder auch Wissensaustausch ist, finden mit dem Chat nur mangelhafte Unterstützung zur textbasierten synchronen Diskussion. Dieses Problem findet sich auch wieder bei der Unterstützung von Telelerngruppen, bei denen es ebenfalls in erster Linie um den Wissensaustausch geht (RÖTTING & BRUDER 2000). Die Diskussion und das damit zusammenhängende kooperative Austauschen und Schaffen von Wissen wird auch in der Literatur (GRÄSEL ET AL. 1997, DÖRING 1997) als einer der Schlüssel zum Lernen gesehen. Im Folgenden werden Chat-Systeme¹ etwas eingehender und vor allem im Kontext des Lernens diskutiert. Eine Auseinandersetzung mit den Schwächen bestehender Systeme schließt sich an.

Chats sind dazu gedacht, textbasierte, synchrone und verteilte Kommunikation in Gruppen zu unterstützen. Sie sind im Allgemeinen nicht spezialisiert. Auf der einen Seite können mit dieser Infrastruktur so beliebige Themen diskutiert werden, auf der anderen Seite bieten Chats auch keine themenspezifischen Unterstützungstechniken an.

Weiterhin verläuft die Kommunikation innerhalb eines Chats weitgehend unstrukturiert: Jeder kann etwas in den Chat schreiben, das die anderen lesen können. Unterschiedliche Rollen sind nur wenig ausgeprägt. Vor allem bieten Chats normalerweise keine Unterstützung für die Führung bestimmter Gesprächsformen an.

Schließlich sind Chat-Systeme bezüglich ihrer Grundfunktionalität sehr einfach zu bedienen. Nach nur geringer Einarbeitungszeit sind die meisten Nutzer in der Lage, einen Chat aus technischer Sicht zu bedienen. Die sinnvolle Nutzung, die auch das Erlernen bestimmter Verhaltensweisen mit einbezieht, ist allerdings erst nach längerem Umgang mit dem System möglich.

¹ Chat-Systeme und MUDs (Multi User Dungeon – siehe beispielsweise <http://autos.cs.tu-berlin.de/~tubmud/>) unterscheiden sich aus technischer Sicht nicht. Die Unterschiede werden hier durch die verwendeten Metaphern und die Herkunft – MUDs stammen aus dem Bereich der Rollenspiele – geprägt. Aus diesem Grund wird auf MUDs in diesem Aufsatz nicht näher eingegangen.

Schwächen

Die meisten Chat-Systeme kranken bisher an verschiedenen Punkten und behindern so den effizienten Wissensaustausch und schränken das Erschaffen von neuem Wissen ein. Diese Schwächen sollen im folgenden aufgeführt und kurz erläutert werden.

Ein Schwachpunkt für die Nutzung eines Chats für ernsthafte Diskussionen, wie sie in Lernszenarien vorkommen, sind fehlende Protokollierungsfunktionen. Häufig kommt es vor, dass einzelne Teilnehmer später in eine Diskussion einsteigen. In solchen Fällen ist es hilfreich, bisherige Beiträge rückwirkend abrufen zu können, um darauf basierend aktiv an der Diskussion teilzunehmen. Aber auch für spätere Zeiten sollten Protokolle einer Diskussion verfügbar gemacht werden. Bisher geschieht dies fast ausschließlich Client-seitig.

Weiterhin existieren bisher keine Chat-Systeme, die eine moderierte Diskussion technisch unterstützen², obwohl verschiedene Autoren wie auch KOPPENHÖFER ET AL., (2000) auf deren Wichtigkeit hinweisen. Wie in klassischen Face-to-Face-Gesprächen sollte ein Moderator Rederecht erteilen, Beiträge zurückstellen oder bisherige Ergebnisse zusammenfassen können.

In Chat-Systemen findet sich keine technische Unterstützung für Bezugnahmen. Es ist nicht möglich Bezug auf vorher gemachte Beiträge zu nehmen, es sei denn, man weist in seinem eigenen Beitrag explizit darauf hin. Verbindungslinien oder farbige Markierungen können helfen, um Bezüge optisch leichter wahrzunehmen und so Missverständnisse zu vermeiden. In Newgroups werden Bezüge derzeit schon als Threads in einer Baumstruktur dargestellt. Allerdings sind hier nur neue Verzweigungen möglich, wohingegen integrierende Beiträge, die das Ziel haben, einzelne Äste der Diskussion zusammenzuführen, nicht darstellbar sind.

Darüber hinaus ist auch die Darstellungsfähigkeit der meisten Chat-Systeme derart unzureichend, dass sie im Kontext des Gruppenlernens nur schlecht eingesetzt werden können. So erlauben nur wenige Systeme eine Darstellung der Textbeiträge, die über eine zeitlich sortierte sequentielle Liste hinausgeht. Im graphischen Aufbau unterscheiden sich alle Chat-Systeme nur unwesentlich. Kernstück ist immer ein Textfeld, in dem Beiträge in chronologischer Reihenfolge dargestellt werden. Auf diese Weise ist – wie auch schon weiter oben angemerkt – häufig schwer ersichtlich, in welchem Bezug Beiträge zueinander stehen. Eine Erweiterung um ein Whiteboard, wie es auch KOPPENHÖFER ET AL. (2000) bei der Sitzungsunterstützung von Lerngruppen für sinnvoll halten und das eng mit dem Chat verzahnt ist, kann evtl. helfen, um das Ergebnis der Diskussion schon im Verlauf evolvierend festzuhalten.

Schließlich ist es kaum möglich, innerhalb des Chats zusätzliche Kontextinformationen in die Diskussion einzubringen. In wissensintensiven Diskussionen ist es oft hilfreich, den Gegenstand der Diskussion darzustellen. Dies ist zwar nicht in allen Fällen möglich, oft jedoch werden Themen erörtert, die sich auf Artefakte beziehen, die mit anderen Applikati-

² IRC verwendet in der Spezifikation den Begriff des Moderators. Dieser wird hier jedoch nur im technischen Sinne verstanden als Verwalter eines Channels (themenspezifische Chat-Gruppe), der je nach Voreinstellung Interessierte zur Diskussion einladen kann oder auch bei Notwendigkeit Teilnehmer aus der Gruppe ausschließen kann.

onen innerhalb des Computer-Systems erstellt worden sind (z.B. Textverarbeitungen, Graphikprogramme etc.). Dieses Problems haben sich auch CHURCHILL ET AL., (2000) angenommen, die zu diesem Zweck eine Systemerweiterung entwickelten, die es ermöglicht an einzelne Absätze eines Dokuments innerhalb einer Textverarbeitung einen Chat anzuhängen zur Diskussion des in dem Absatz behandelten Themas. Allerdings ist hier der Kontext immer auf genau einen Absatz beschränkt, es können nicht beliebige Artefakte mit einem Chat verknüpft werden. Weiterhin wird auf diese Weise nicht das Problem sondern das Artefakt in den Mittelpunkt des Interesses gerückt.

2.2 Aktivität und Interaktion

Ein wesentliches Merkmal der konstruktivistischen Lerntheorie ist umfangreiche Aktivität des Lernenden. Erst aktive und expressive Tätigkeiten des Lernenden, insbesondere direktes Lernen in einer Arbeitsumgebung und eigenes Problemlösen, fördern die Verinnerlichung erworbener Fertigkeiten (STRZEBKOWSKI & KLEEGERG 2002). An gleicher Stelle werden über einfache Steuerungsinteraktionen hinausgehende sinnvolle didaktische Interaktionsformen erst dann als gegeben angesehen, „wenn diese bei den Lernenden

- aktives Denken und intensive Elaborationsprozesse auslösen,
- expressive und kreative Tätigkeiten zulassen und fördern
- zum einsichtsvollen, bedeutungsvollen und entdeckenden Lernen führen.“

Sinnvolle didaktische Interaktionen bestehen nach STRZEBKOWSKI & KLEEGERG (2002) z. B. in der

- Modifikation oder Anpassung vorhandener Daten und Lernwege
- Kreation neuer multimedialer Daten oder Objekte
- Nutzung eines elektronischen Notizblocks

Die Umsetzung einer Programmierumgebung lässt offensichtlich ein sehr hohes Maß an Aktivität und Interaktion zu, bietet sie doch über einfache Steuer- und beschränkte Eingabemöglichkeiten hinaus die weitgehend freie Betätigung des Lernenden.

2.3 Visualisierung, Algorithmenanimation

Erst die explizite Visualisierung ermöglicht eine konkrete Vorstellung abstrakter, interner Programmabläufe. So verwenden Lehrbücher üblicherweise Abbildungen, um einzelne Zustände eines Algorithmus darzustellen (OTTMANN & WIDMAYER 1996). Durch die Möglichkeiten von Bewegtbildern in Form von Filmen oder Animationen ergeben sich besonders für komplexere Visualisierungen, die sonst nur unzulänglich dargestellt werden könnten, wichtige Gestaltungsmöglichkeiten (WEIDENMANN 2002a). Für Algorithmen ist damit eine weitaus differenziertere Darstellung – einschließlich der Visualisierung von Zustandsänderungen – möglich. Eine solche Animation von Algorithmen und Datenstrukturen ist seit vielen Jahren ein Bereich intensiver Forschung, und seither wurde eine Vielzahl von Systemen für verschiedene Programmiersprachen entwickelt. Bekannte Beispiele sind LEONARDO (CRESCENZI ET AL. 1997), Mocha (BAKER ET AL. 1996) und JELIOT (BEN-ARI ET AL. 2002); einen Überblick geben z.B. KERREN & STASKO (2002).

Die Animation erfolgt dabei meist über die Einbindung von Visualisierungsanweisungen in den Quellcode des Algorithmus. Gängige Techniken sind die der „Interested Events“ und des „State Mapping“. Bei den „Interested Events“ identifiziert der Programmierer eines Algorithmus wichtige Anweisungen im Programm und bindet an diesen Stellen Visualisierungsaufrufe ein; beim „State Mapping“ wird nach jeder Änderung eines Variablenwerts der aktuelle Programmzustand in Abhängigkeit von diesen Variablen visualisiert (DEMETRESCU, FINOCCHI & STASKO 2002).

Es ist zu beobachten, dass viele Systeme zwischen Programmierer und Nutzer differenzieren, d.h. der Programmierer hat einen Programmablauf erstellt, für den Nutzer tritt lediglich das visualisierte Ergebnis in Erscheinung. Eine Interaktion ist oft nur über die Eingabe einzelner Parameter über Eingabefelder möglich. Aus Sicht des Konstruktivismus (s.o.) ist es dagegen vorteilhafter die Trennung zwischen Programmierer und Nutzer aufzuheben, d.h. der Nutzer erstellt Programmablauf und Visualisierung. Die daraus resultierende Multicodierung durch Text (Programmcode) und Bild (Animation) hat einen weiteren positiven Einfluss auf den Wissenserwerb des Lernenden (WEIDEMANN 2002b).

Die Effizienz von Algorithmenanimationen in der Ausbildung wurde aufgrund widersprüchlicher Evaluationsergebnisse oft angezweifelt (HUNDHAUSEN, DOUGLAS & STASKO 2002), allerdings wird an gleicher Stelle in einer Meta-Studie über die Ergebnisse von 24 Evaluationen festgestellt, dass Systeme, die den Lernenden aktiv einbeziehen, sehr wohl das Erlernen von Algorithmen effektiv und erfolgreich unterstützen.

2.4 Entzerrung der Komplexität

Für erste Programmierschritte zum Erlernen grundsätzlicher Funktionen und zur Umsetzung einfacher Algorithmen genügt eine einfache Entwicklungsumgebung, die lediglich einfache Möglichkeiten zur Steuerung von Programmen vorsieht. Zusätzlich sollte der Anfänger von den syntaktischen Besonderheiten einer komplexen Programmiersprache befreit werden: Alle in einem Programm verwendeten Elemente sollten von Beginn an bekannt und verstanden sein. Ein solches Konzept für Schüler beschreiben mit Kara to Java z. B. REICHERT, NIEVERGELT & HARTMANN (2000).

2.5 Zusammenfassung

Obwohl es in einzelnen der angeführten Bereiche eine Vielzahl bestehender Systeme gibt, integriert keines alle Funktionalitäten – Kommunikation, Visualisierung, Programmierumgebung und vereinfachte Programmiersprache auf Basis von Java – in einem System.

Diese Lücke schließt das GeoCafé. Die Basis bildet eine hochgradig flexible Kommunikationsplattform für das gemeinsame Lernen in verteilten Arbeitsgruppen. Als technische Grundlage dient eine verteilte Plattform für komponentenbasierte Applikationen, auf die im folgenden Kapitel im Detail eingegangen wird. Für das Erlernen von Geo-Algorithmen und –Datenstrukturen wurden als spezifische Komponenten eine kooperative Programmierumgebung und ein Geovisualisierungsbaustein für den explorativen und kooperativen Umgang mit Algorithmen realisiert. Diese werden in Kapitel 4 erläutert.

3 Kommunikation

Das GeoCafé soll als modular aufgebaute Kommunikationsplattform die Basis für das gemeinsame Lernen von Inhalten im Bereich Geoinformation bilden. Sie ist als Client-Server-Anwendung konzipiert. Abbildung 1 gibt eine Übersicht über die wichtigsten Funktionen des GeoCafé. Sie sollen im Folgenden kurz angesprochen werden:

- **Benutzer- und Benutzergruppenverwaltung:** Das GeoCafé stellt eine eigene Benutzerverwaltung (siehe Abbildung 1, oben links) zur Verfügung. Angemeldete Benutzer müssen sich beim Start ihres Clients authentifizieren. Es ist weiterhin möglich, Rollen zu unterscheiden. Sie werden gemeinsam mit einem selbst änderbaren Profil gespeichert. Die Rollen sind aus technischer Sicht entscheidend für die Darstellung des GeoCafé am Client und die verfügbaren Funktionalitäten. Für die Arbeit in der Lerngemeinschaft dienen sie auch zur Festlegung von Funktionen innerhalb einer Lerngruppe. Beispielsweise kann ein Mitglied die Rolle „Moderator“ einnehmen. Seine Textbeiträge werden dann besonders kenntlich gemacht.

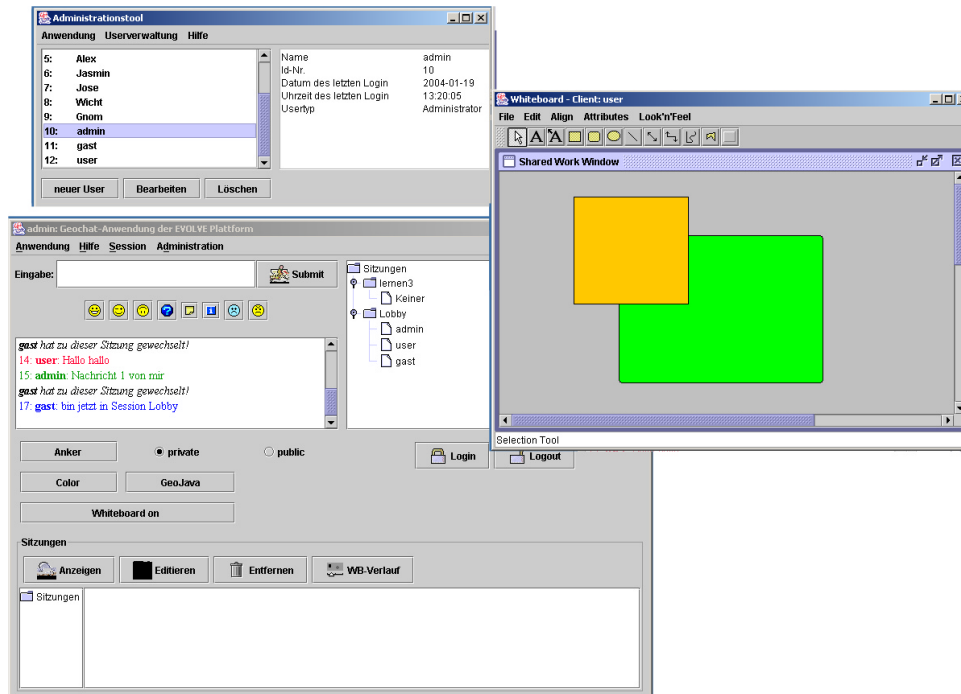


Abb. 1: Benutzer- und Sitzungsverwaltung, Whiteboard und Chat des GeoCafés

- **Integration eines Shared Whiteboard:** Das Whiteboard, eine Zeichenfläche, die von allen Teilnehmern einer Gruppe gleichzeitig beschrieben werden kann, stellt über den (Chat)-Text hinaus ein Medium zur Kommunikation zur Verfügung. So dient das Whiteboard zum einen zur Erweiterung der Darstellungsmöglichkeiten (Skizzen), zum an-

deren lassen sich aber auch diskussionswürdige Artefakte mit anderen Gruppenmitgliedern „teilen“ (vorhandene Graphiken, Screenshots etc.).

- **Unterstützung von Sprechakten durch Bezugnahme:** Ein großes Manko bisheriger Chat-Systeme ist die mangelnde Unterstützung des Sprechaktes. Textbeiträge werden sequentiell ohne direkten Bezug zueinander dargestellt. Solche Bezüge müssen dann sprachlich formuliert werden (z. B. „Wie Du oben schon sagtest...“). Der GeoChat als Teil des GeoCafé unterstützt solche Bezugnahmen durch die Möglichkeit der expliziten Referenzierung. Neben dem üblicherweise vorhandenen Button „Submit“, mit dem die Textabgabe abgeschlossen und der Beitrag gesendet wird, gibt es weiterhin die Möglichkeit zur Erzeugung eines Ankers auf ein beliebiges Element. Dies kann sowohl ein früherer Textbeitrag als auch das Whiteboard (in seinem aktuellen Zustand) sein. Solche verankerten Beiträge werden im Chat als solche gesondert visualisiert. Alle Benutzer können dann in ihrem Chat-Client auf eine solche Referenz klicken und der Anker wird dargestellt.
- **Protokollierung:** Im GeoCafé werden standardmäßig alle Chat-Sessions, von denen es mehrere gleichzeitig geben kann, protokolliert. Dabei dient das Protokoll einer laufenden Session als Nachschlagefunktion. Darüber hinaus können später hinzukommende Benutzer den bisherigen Stand der Session einsehen und so schnell in die Diskussion finden. Speziell für das Whiteboard, in dem Elemente nicht nur iterativ gezeichnet, sondern auch während einer Session gelöscht werden können, ist diese Funktion sehr wertvoll. Die gesamte Session (oder auch nur Ausschnitte daraus) kann wiederholt dargestellt werden. Session-Protokolle werden nach Beendigung auf dem Server gespeichert und können bei Bedarf ausgedruckt werden. So bleiben Diskussionen über einen längeren Zeitraum erhalten und können zum Selbstlernen wiederholt betrachtet werden.
- **Anpassbarkeit des GeoCafé und Modularität:** Das gesamte GeoCafé ist modular aus unabhängigen Einzelkomponenten aufgebaut. Dies bietet völlig neue Möglichkeiten in Bezug auf die Anpassbarkeit der Client-Applikation: Für die Administratoren bietet es die Möglichkeit, Komponenten je nach Rolle der Benutzer verfügbar zu machen oder auszublenden. Die Benutzer selbst haben darüber hinaus die Möglichkeit, den eigenen Client entsprechend ihren Bedürfnissen zu konfigurieren. Solche Anpassungsmöglichkeiten gehen weit über die Größe und Position der Einzelkomponenten am Bildschirm hinaus. Es ist auch möglich, die lokale Protokollierungsfunktion zu entfernen oder mehrere Whiteboards parallel zu integrieren. Die technische Realisierung dieser komponentenbasierten Anwendung ist Thema des nächsten Abschnitts.

FREEEVOLVE als flexible Plattform für ein erweiterbares GeoCafé

Die FreEvolve-Plattform (STIEMERLING 2000) ist eine client-server-basierte Umgebung, in der verteilte komponentenbasierte Applikationen ablaufen können. Dazu verwaltet der FreEvolve-Server alle verfügbaren Komponenten. Wählt ein Benutzer eine Applikation aus, so wird ihr Plan, der die benötigten Komponenten und ihr Zusammenspiel beschreibt, geladen und analysiert. Anschließend werden die benötigten Komponenten auf dem Server und dem Client instanziiert und entsprechend dem Plan miteinander verbunden. Das GeoCafé wird also durch eine Sammlung von FREEEVOLVE-Komponenten und dazugehörigen Plänen realisiert.

In unserem Kontext ist die FREEEVOLVE-Plattform vor allem deswegen besonders geeignet, weil sie Benutzerprofile auf dem Server verwaltet. Auf diese Weise können sich die Ler-

nenden von unterschiedlichen Clients aus einloggen und erhalten dennoch ihre gewohnte Umgebung. Darüber hinaus lassen sich für jeden Benutzer und jede Rolle Applikationspläne festlegen, so dass darauf basierend eine Individualisierung erfolgen kann.

Weiterhin bietet die FREEVOLVE-Plattform die Möglichkeit mit Hilfe eines speziellen Anpassungswerkzeugs (WON UND WULF 2003), zur Laufzeit Änderungen an den Kompositionen vorzunehmen (Komponenten hinzuzufügen, zu entfernen oder deren Parameter und Verbindungen zu ändern). Auf diese Weise lässt sich das GeoCafé dann entsprechend den jeweiligen Erfordernissen anpassen. Dies ist besonders für die Administratoren interessant, die mit Hilfe des so genannten TailorClient, der die Anpassungsvorgänge durch Prüfung der Integrität unterstützt (WON UND CREMERS 2003), für Benutzergruppen spezielle GeoCafé-Kompositionen vorfertigen können.

4 Programmierung und Visualisierung von Algorithmen

Für den explorativen Umgang mit Geo-Algorithmen und –Datenstrukturen wurde die oben beschriebene Kommunikationsplattform um vier Komponenten zu einer kooperativen Programmierumgebung erweitert. Für den Nutzer sichtbar sind ein Programmeditor, ein Visualisierungsfenster und eine Bedienleiste. Eine weitere Komponente steuert im Hintergrund den Ablauf der vom Nutzer erstellten Programme und deren Visualisierung.

4.1 Ablauf einer Lernsitzung

Die Programmierumgebung des GeoCafés stellt Lernenden einen virtuellen Lernraum zur Verfügung, der die Entwicklung und Ausführung von Programmen alleine oder im Team erlaubt. Abbildung 2 zeigt schematisch die Konstellation einer Zweiergruppe; eine Hinzunahme weiterer Gruppenmitglieder ist jederzeit möglich. In die Benutzeroberfläche integriert sind Benutzerverwaltung, Chat und Whiteboard (vgl. Kapitel 3) sowie Bedienleiste, Editor und Visualisierung. Die für einen Nutzer gesperrten Bereiche sind rot gefärbt, in grün gefärbten besitzt er Schreibrechte.

Als typisches Anwendungsszenario in der Konstellation einer Zweiergruppe wird an dieser Stelle in Kürze die gemeinsame Entwicklung eines Programms skizziert. Nutzer 1 (N1) besitzt Schreibrecht im Editor, über die Bedienleiste ist er berechtigt Programme auszuführen. Die Visualisierung ist während der Programmentwicklung zunächst inaktiv. Für Nutzer 2 (N2) sind dagegen Editor und Bedienleiste gesperrt, es werden aber alle Eingaben von N1 synchron bei N2 dargestellt. Über Chat und Whiteboard ist jederzeit eine Kommunikation über die Inhalte möglich. N2 ist erst dann berechtigt Editor und Bedienleiste zu nutzen, wenn die Schreibrechte auf ihn übergegangen, und die entsprechenden Komponenten bei N1 gesperrt sind. Der Übergang der Schreibrechte erfolgt im Dialog der beteiligten Nutzer; fordert ein Nutzer Schreibrechte an, kann sie der jeweilige Inhaber abtreten. Beim Ausführen eines Programms wird der Ablauf sowohl synchron auf beiden Systemen, als auch synchron in den Komponenten „Editor“ – durch Hervorheben der aktuellen Programmzeile – und „Visualisierung“, dargestellt. Während einer Sitzung hat ein Übungsleiter jederzeit die Möglichkeit sich ein- oder auszuschalten und an der Diskussion teilzunehmen.

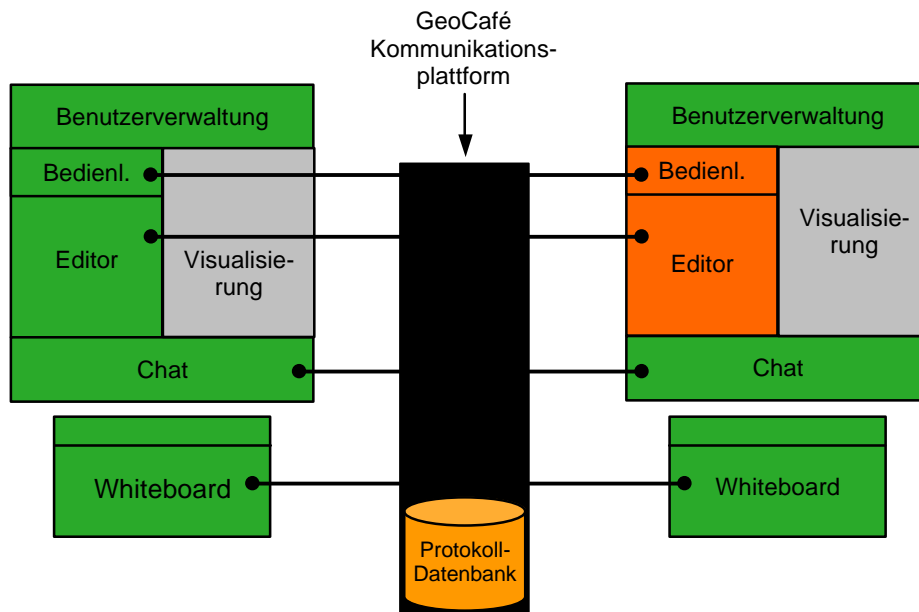


Abb. 2: Logische Darstellung der Programmierumgebung des GeoCafés auf zwei vernetzten Arbeitsplätzen; rote Bereiche sind für den jeweiligen Nutzer gesperrt, in grünen Bereichen besitzt er Schreibrechte

Neben der Protokollierung der Kommunikation (Kapitel 3) werden analog zum Whiteboard und Chat auch alle Eingaben im Editor protokolliert. Eine Sitzung ist so im Nachhinein lückenlos, z.B. für einen Übungsleiter, rekonstruierbar. Darüber hinaus erlaubt es die Protokollierung in der Zeit rückwärts zu springen: In einer Sitzung ist es jederzeit möglich – beispielsweise bei einer fehlerhaften Programmierung – die gesamte Lernumgebung auf einen fehlerfreien Status in der Vergangenheit zurückzusetzen und von dort eine neue Lernsitzung zu starten.

4.2 Komponenten

GeoJava Editor: Der GeoJava Editor ist ein einfacher Texteditor, der sich an den für Programmeditoren üblichen Funktionen orientiert, im Funktionsumfang aber stark eingeschränkt ist. Angepasst an die Erfordernisse von GeoJava, der Programmiersprache des GeoCafés, bietet er ein darauf abgestimmtes „Syntax-Highlighting“, ein „Klammer Matching“ und die Möglichkeit, die aktuell auszuführende Zeile beim Programmablauf hervorzuheben.

GeoJava selbst ist ein Java-Dialekt, der auf Programmieranfänger zugeschnitten ist: Hauptforderung für die Definition der Sprache war, dass Programmierneulinge von Beginn an in die Lage versetzt werden, grundlegende Syntax-Elemente (Schleifen, Bedingungen) „spielerisch“ in Quellcode umzusetzen ohne syntaktische Besonderheiten oder Konzepte der Objektorientierung kennen zu müssen. Dazu wird einerseits der Sprachumfang im Ver-

gleich zu Java vermindert, andererseits aber auch erweitert. So ist es beispielsweise nicht notwendig, für die Ausführung einfacher Programme eine Klasse zu definieren. Modifier wie „public“, „private“ etc. und import-Anweisungen sind ebenso wenig erforderlich; als ausführbare Hauptfunktion genügt „main()“ statt „public static void main(String args[...])“. Die Erweiterung umfasst im wesentlichen vereinfachte Funktionen zur Bildschirmausgabe bzw. Einlesen über die Tastatur sowie eine direkte Nutzbarmachung der Methoden der Klasse „Math()“ ohne explizit den Klassennamen verwenden zu müssen, wie z. B. bei „Math.sin“. Um ein GeoJava-Programm kompilieren und ausführen zu können, wird es zunächst über eine Code-Transformation in Java-Syntax konformen Quellcode übersetzt. Dabei wird vor jeder Anweisung ein virtueller Haltepunkt eingefügt, der es ermöglicht, Programme später an diesen Stellen anzuhalten und so schrittweise auszuführen. Dieses Vorgehen erlaubt es, ganz gezielt die Ausführung des GeoJava-Programms anzuhalten; die Nutzung des Java-Debug-Mode wie z. B. in Netbeans oder Sun ONE Studio (NETBEANS 2004, SUN 2004), hätte mit dem Stop der Programmausführung alle Threads – und damit auch die Komponenten des GeoCafés – angehalten. Die Haltepunkte enthalten zudem eine Verknüpfung zum ursprünglichen GeoJava-Code; so lassen sich eventuelle Fehler zuordnen und die aktuelle Programmzeile beim späteren Ablauf hervorheben.

Zu betonen ist, dass diese syntaktischen Festlegungen besonders Anfängern den Einstieg erleichtern sollen. Um keine Hürden für den späteren Umstieg auf Java aufzubauen, ist ein „fließender Übergang“ möglich, d. h. es können mit dem Lernfortschritt zuvor weggefallene oder erweiterte oder Sprachelemente wie Klassen oder Modifier in GeoJava eingeführt werden. Somit bietet das GeoCafé auch dem fortgeschrittenen Nutzer die Möglichkeit in syntaktisch korrektem Java zu programmieren und den Ablauf zu visualisieren.

GeoJava Bedienleiste: Über die GeoJava Bedienleiste steuert der Benutzer den Ablauf seines Programms. Im Vergleich zu professionellen Entwicklungsumgebungen sind die Funktionen auf ein Minimum beschränkt. Neben Menüpunkten zum Anlegen neuer Dateien und zum Öffnen von Visualisierungsfenstern gibt es lediglich die Möglichkeit ein Programm entweder im Einzelschrittmodus oder durchgängig ablaufend zu Starten, zu Stoppen, eine Pause zu machen und einen Schritt zurück bzw. an Anfang oder Ende zu springen. Über einen Schieberegler ist die Ablaufgeschwindigkeit steuerbar.

GeoJava Visualisierung: Die Visualisierungskomponente zeigt den Ablauf eines Programms visuell an. Dabei wird zwischen zwei Arten von Datenstrukturen, den freien und eingebetteten unterschieden. Als freie Datenstrukturen werden diejenigen bezeichnet, die keinen direkten Geobezug besitzen, wie z. B. Bäume, Heaps, Queues etc.; ihre Platzierung hängt lediglich von grafischen Gesichtspunkten ab. Eingebettete Datenstrukturen, beispielsweise Wegenetzgraphen, sind dagegen georeferenziert und werden über ihre Koordinaten in der Anzeige platziert.

Für die Erfordernisse des GeoCafés werden alle darzustellenden Strukturen auf die grafischen Primitive Kreis, Rechteck, Dreieck, Linie (und Polygon) zurückgeführt. Die Realisierung der Visualisierung erfolgt nach dem Ansatz der „Interested Events“ (Kapitel 2.4) über Einbindung von Anweisungen in das GeoJava-Programm, so zeichnet z. B. „createCircle(double x, double y)“ einen Kreis mit den Koordinaten x und y. Alle Primitive besitzen eine eigene ElementID und können so jederzeit einzeln angesprochen, und insbesondere auf einen Mausclick des Nutzers hin selektiert und hervorgehoben werden.

Für die grafische Darstellung einiger vordefinierter Datenstrukturen, z. B. Binärbäume, ist ein Autolayout implementiert, d. h. nach jedem Einfügen oder Löschen eines Knotens wird die Anordnung automatisch bestmöglich im Visualisierungsfenster angepasst. Bei Bäumen gibt es weiterhin die Möglichkeit, einzelne Teilbäume zu kollabieren, um so eine übersichtlichere Darstellung zu erhalten.

Die Visualisierung erfolgt im Vektorformat SVG (Scaleable Vector Graphics, W3C 2004). Neben einem stufenlosen Zoomen ist besonders von Vorteil, dass SVG einen einfachen Grafikexport sowohl in SVG als auch in Rasterformaten möglich macht; statische Programmzustände sind so einfach zu visualisieren und in andere Medien, insbesondere in das oben beschriebene Whiteboard, integrierbar. Basis der Komponente ist das Open Source Projekt Batik (BATIK 2004), eine Java-Implementation eines SVG – Viewers. Batik wurde an die Erfordernisse des GeoCafés angepasst, allerdings nicht auf diese beschränkt, sondern als erweiterbare, modular einsetzbare Anzeigekomponente auch für andere Programme ausgelegt.

GeoJava Steuerung: Die Steuerung koordiniert im Hintergrund die sichtbaren Abläufe der Programmierumgebung. Über diese Komponente werden transformierte GeoJava-Programme geladen und schrittweise ausgeführt. Editor und Visualisierungsfenster müssen bei dieser Komponente registriert sein; beim Ablauf von GeoJava-Programmen erhält dann der Editor Informationen über die aktuell hervorzuhebende Zeile, die Visualisierung über Grafik-Events im Programm.

Die Zusammenstellung dieser Komponenten erfolgt zur Ausführungszeit. So hat ein Nutzer die Möglichkeit, mehrere Programme mit mehreren Visualisierungsfenstern zur gleichen Zeit geöffnet zu haben oder verschiedene Algorithmen auf der gleichen Datengrundlage auszuführen und entweder in verschiedenen oder im gleichen Anzeigefenster darzustellen, beispielsweise die Berechnung der konvexen Hülle und der Delaunay-Triangulation auf einer Menge von Punkten.

5 Zusammenfassung und Ausblick

In diesem Beitrag wurde die kooperative Lernumgebung „GeoCafé“ vorgestellt, ein virtuellen Raum, der durch eine kohärente Einführung in die Programmierung sowie Algorithmen und Datenstrukturen das konstruktivistische Erlernen informatischer Inhalte allein und in der Gruppe unterstützt. Erste praktische Experimente haben sich als sehr vielversprechend erwiesen, zum Zeitpunkt des Verfassens dieses Textes steht eine umfassende Evaluation allerdings noch aus.

Für die Zukunft sind über die Laufzeit des BMBF-Projekts hinaus folgende Erweiterungen denkbar bzw. geplant: Es hat sich gezeigt, dass gerade für den Einsatz in Lehrveranstaltungen ein Werkzeug, das einem Gruppenleiter die Vordefinition von Gruppen in physikalischen Lehrräumen wie z. B. Rechnerpools, gestattet, wünschenswert ist. Weiterhin wird das GeoCafé in naher Zukunft um Klassenbibliotheken für GeoJava erweitert. Diese sollen u. a. vorgefertigte Datenstrukturen mit eingebauter Visualisierung enthalten, die direkt einsetzbar sind. Beispiele sind Lineare Liste, Binärer Suchbaum, AVL-Baum, Heap, B-Baum, R-Baum und Quadtree.

Literatur

- Baker, J. E., Cruz, I. F., Liotta, G. & Tamassia, G. (1996): Algorithm Animation Over the World Wide Web. In: Proc. Int. Workshop on Advanced Visual Interfaces (AVI '96).
- Ben-Ari, M., N. Myller, E. Sutinen & J. Tarhio (2002): Perspectives on Program Animation with Jeliot. In: Diehl, S. (Ed.): Software Visualization. Springer-Verlag Berlin Heidelberg.
- Churchill, E. F., J. Trevor, S. Bly, L. Nelson & D. Cubranic (2000): "Anchored Conversations: Chatting in the context of a document," in: Proceedings of CHI 2000, The Hague, Amsterdam, ACM, 454-46.
- Crescenzi, P., C. Demetrescu, I. Finocchi & R. Petreschi (1997): Leonardo: a software visualization system. In: Proceedings of the 1st Workshop on Algorithm Engineering (WAE'97).
- Demetrescu, C., I. Finocchi & J. T. Stasko (2002): Specifying Algorithm Visualizations: Interesting Events or State Mapping? In: Diehl, S. (Ed.): Software Visualization. Springer-Verlag Berlin Heidelberg.
- Döring, N. (1997): „Lernen und Lehren im Internet“. In: Batinic, B. (Ed.): Internet für Psychologen. Hofgrewé-Verlag, S. 359-388.
- Gräsel, C., F. Fischer, J. Bruhn, & H. Mandl (1997): „Ich sag Dir was, was Du schon weißt. Eine Pilotstudie zum Diskurs beim kooperativen Lernen mit Computernetzen. Forschungsbericht Nr. 82,“ Ludwig-Maximilians-Universität, Lehrstuhl für Empirische Pädagogik und Pädagogische Psychologie, München.
- Hundhausen, C. D., S. A. Douglas & J. T. Stasko (2002): A Meta-Study of Algorithm Visualization Effectiveness. Journal of Visual Languages and Computing, Vol. 13, No. 3
- Kerren, A. & J. T. Stasko (2002): Algorithm Animation. In: Diehl, S. (Ed.): Software Visualization. Springer-Verlag Berlin Heidelberg.
- Kopp, B., M. Zabel, & H. Mandl (2001): Dozentenleitfaden für die mediendidaktische Gestaltung problemorientierter virtueller Lernumgebungen an Hochschulen. München: Ludwig-Maximilians-Universität, Department Psychologie, Institut für Pädagogische Psychologie.
- Koppenhöfer, C., T. Böhm, & H. Krcmar (2000): „Integral - Methodische Integration multimedialer und interaktiver Lernwerkzeuge zur Optimierung der Gestaltungskompetenz in der arbeitswissenschaftlichen Lehre“. In: Proceedings of D-CSCL 2000, Vernetztes Lernen mit digitalen Medien, Darmstadt, Deutschland, Physica-Verlag, 147-162.
- Ottmann, T. & P. Widmayer (1996): Algorithmen und Datenstrukturen. Spektrum Akademischer Verlag, Heidelberg, Berlin, Oxford.
- Reichert, R., J. Nievergelt & W. Hartmann (2000): Ein spielerischer Einstieg in die Programmierung mit Java. Informatik Spektrum, Band 23, Heft 5
- Reinmann-Rothmeier, G. & H. Mandl (2001). Unterrichten und Lernumgebungen gestalten. In: A. Krapp & B. Weidenmann (Hrsg.), Pädagogische Psychologie. Ein Lehrbuch. Weinheim: Psychologie VerlagsUnion.

- Rötting, M. & R. Bruder (2000): „Integral - Methodische Integration multimedialer und interaktiver Lernwerkzeuge zur Optimierung der Gestaltungskompetenz in der arbeitswissenschaftlichen Lehre“. In: Proceedings of D-CSCL 2000, Vernetztes Lernen mit digitalen Medien, Darmstadt, Deutschland, Physica-Verlag, 57-52.
- Stiemerling, O. (2000): „The Evolve Project.“: Component-Bases Tailorability. Dissertation am Institut für Informatik III, Universität Bonn.
- Strzebkowski, R. & N. Kleeberg (2002): Interaktivität und Präsentation als Komponenten multimedialer Lernanwendungen. In: Issing, L. J. & P. Klimsa (Hrsg.): Information und Lernen mit Multimedia und Internet. Verlagsgruppe Beltz, Psychologische Verlags Union, Weinheim.
- Weidenmann, B. (2002a): Abbilder in Multimediaanwendungen. In: Issing, L. J. & P. Klimsa (Hrsg.): Information und Lernen mit Multimedia und Internet. Verlagsgruppe Beltz, Psychologische Verlags Union, Weinheim.
- Weidenmann, B. (2002b): Multicodierung und Multimodalität im Lernprozess. In: Issing, L. J. & P. Klimsa (Hrsg.): Information und Lernen mit Multimedia und Internet. Verlagsgruppe Beltz, Psychologische Verlags Union, Weinheim.
- Won, M. & A. B. Cremers (2002): „Supporting End-User Tailoring of Component-Based Software - Checking Integrity of Composition“. In: Proceedings of Colognet 2002 (Conjunction with LOPSTR 2002), Madrid, Spain, 19.-20.09.
- Won, M. & V. Wulf (2003): „Anpassungsumgebung für komponentenbasierte Software: Kooperativ und lernförderlich“. In: Oberquelle, H., W. Prinz & J. Ziegler (Hrsg.): icom - Zeitschrift für interaktive und kooperative Medien, 1/2003, 28-34.

Adressen im WWW

- Batik (2004): Open Source SVG-Viewers Batik. <http://www.apache.org> (zuletzt besucht am 26.01.2004).
- NetBeans (2004): Entwicklungsumgebung netBeans. <http://www.netbeans.org> (zuletzt besucht am 26.01.2004).
- Sun (2004): Entwicklungsumgebung Sun ONE Studio. <http://www.sun.com> (zuletzt besucht am 26.01.2004).
- UML (2004): Unified Modeling Language. <http://www.uml.org> (zuletzt besucht am 26.01.2004).
- W3C (2004): World Wide Web Consortium. <http://www.w3.org> (zuletzt besucht am 01.02.2004)