Expedition in das Datenreich: Exploratives Erlernen von Internetdiensten

Lars Kornelsen, Ulrike Lucke, Djamshid Tavangarian
Universität Rostock, Institut für Informatik, Lehrstuhl für Rechnerarchitektur
{vorname.nachname}@uni-rostock.de

Abstract: Eine intensive, selbstgesteuerte Auseinandersetzung der Lernenden mit einer praktischen Problemstellung bewirkt eine meist tiefere Durchdringung des Themas als bei der passiven Aufnahme von Fakten. Durch Entwicklung von Werkzeugen zur gezielten Unterstützung nicht nur der traditionellen Präsenzlehre, sondern vor allem auch des explorativen Lernens kann demnach eine Steigerung individueller Lernerfolge erreicht werden. Der Beitrag beschreibt ein webbasiertes Werkzeug zur selbstgesteuerten, praktischen Auseinandersetzung mit verschiedenen Internetdiensten. Dabei können wahlweise sowohl reale Dienste als auch Simulatoren kontaktiert werden. Die visuelle Aufbereitung der übertragenen Nachrichten erfolgt in unterschiedlichen Ansichten, z. B. in Baum- oder Nachrichtenform. Der Einsatz des Werkzeugs wird exemplarisch am Domain Name System (DNS) demonstriert.

1 Einleitung

1.1 Grundsätze des explorativen Lernens

Unter dem Begriff des *explorativen Lernens* werden Ansätze der selbstgesteuerten Wissenskonstruktion auf Basis von Versuch und Irrtum zusammengefasst, deren Wurzeln im Konstruktivismus liegen. In einem hochgradig interaktiven, fast spielerischen Prozess löst der Lernende ein unmittelbar in den Anwendungskontext eingebettetes Problem [Ste00]. Der Lehrende tritt dabei in die Rolle eines Beobachters zurück und gibt außer initialen oder gelegentlichen begleitenden Stimuli keine Anweisungen.

Die Experimente müssen nicht unbedingt an einem *realen Objekt* erfolgen. Während für Chemiker etwa regelmäßig der Umgang mit verschiedensten Reagenzien auf der Tagesordnung steht, ist beispielsweise für angehende Mediziner oder Architekten aufgrund möglicher dramatischer Auswirkungen von Fehlern eher eine realitätsnahe *Simulation* zu bevorzugen [Med][Ple04]. Darüber hinaus bietet auch der Fernzugang zu realen Objekten oder Systemen ein hohes pädagogisches Potenzial [FHW04][KKK02].

Die *Vorteile* des explorativen Lernens sind vielfältig. Der direkte Anwendungsbezug wird von Lernenden meist als motivationssteigernd empfunden. Die intensive praktische Auseinandersetzung mit einer realistischen Problemstellung führt zu einem tieferen Verständnis als bei einer passiven Konsumierung von Informationen [NR02]. So kann das erworbene Wissen leichter auf andere Szenarien transferiert werden.

Als grundsätzlicher *Kritikpunkt* am explorativen Lernen wird vor allem die schwierige Prüfbarkeit der erworbenen praktischen Fertigkeiten (im Gegensatz zu abfragbarem Faktenwissen) angeführt. Der individuelle Lernerfolg ist zudem stark von der intrinsischen

Motivation der Lernenden abhängig. Oft klaffen erlernte Fähigkeiten bei inhomogenem Vorwissen anschließend noch weiter auseinander. In der Praxis scheitert das Konzept des explorativen Lernens daher oft an organisatorischen Fragen wie Größe und Heterogenität der Lernergruppen [Sch03]. Die immaterielle Natur des Fachgebiets Informatik bietet allerdings gerade im Kontext des eLearning gute Ausgangsbedingungen für derartige Szenarien, wie nachfolgend am Beispiel von Internetdiensten gezeigt werden soll.

1.2 Internetdienste aus didaktischer Sicht

Die Vielfalt und Komplexität existierender Internetdienste (zum Beispiel Verzeichnis-, Datentransfer- oder Namensdienste) sind Gegenstand zahlreicher Lehrveranstaltungen in der Informatik. Während theoretische Abhandlungen oder Demonstrationen durch den Lehrenden nur bedingt geeignet sind, ein tieferes Verständnis der Zusammenhänge beim Lernenden zu wecken, bergen Praktika ein deutlich höheres Potential, jedoch auch einen erheblich höheren technischen und logistischen Aufwand.

Didaktisch motivierte *Werkzeuge* zur Unterstützung des explorativen Lernens sind in der Informatik bislang weniger verbreitet. Die Entwicklung von Software, Datenbanken oder Computergraphiken wird meist am „lebenden“ Rechner erlernt; die Studierenden werden sozusagen „ins kalte Wasser geworfen“. Computersysteme in ihrer bestehenden Komplexität sind auf diese Weise jedoch nicht immer hinreichend intuitiv erfahrbar. Deshalb ist eine Einbettung des betrachteten Objekts in eine Experimentierumgebung sinnvoll, die Wesentliches hervorhebt, Nebensächliches in den Hintergrund stellt und durch kontextuelle Einbettung den Lernprozess gezielt unterstützt. Dies bieten gerade bei Internetdiensten die derzeit eingesetzten, meist kommandozeilenbasierten Programme wie z. B. `nslookup` jedoch nicht. Angesichts der wachsenden Bedeutung der Informatik auch als Baustein für die Ausbildung in anderen Fächern müssen höhere Abstraktionsniveaus gefordert werden.

Geeignete *Visualisierungen* sind ein wichtiges Element solcher Umgebungen [NR03]. Sie können nicht nur im Rahmen des selbstgesteuerten, explorativen Lernens eingesetzt werden, sondern stellen auch eine sinnvolle Bereicherung des Frontalunterrichts dar. Gerade bei Internetdiensten ermöglichen Visualisierungen der untersuchten Protokolle, Strukturen oder Prozesse eine deutlich anschaulichere und abstraktere Demonstration der Funktionsweise als dies mit der rein nachrichtenbasierten Betrachtung von Requests und Responses möglich ist.

Die rechnergestützte Ausbildung umgeht zudem einige mit dem explorativen Lernen ansonsten verbundene Nachteile. Digitale Umgebungen sind im Gegensatz zu materiellen Experimenten nahezu beliebig replizierbar und dank Zugang über das Internet auch bei zahlreichen oder entfernten Lernenden kein Problem. Beim Einsatz von Simulatoren für Softwaresysteme verschwinden zunehmend die Grenzen zwischen realem und virtuellem Experiment. Die Versuche können individuell parametrisiert werden und weisen somit für heterogene Zielgruppen eine hohe inhärente Vielfalt unter Wahrung der Konsistenz auf. Durch eine Kopplung an bestehende Lehr- und Lernmaterialien wird zudem direkt im Lernprozess eine Homogenisierung des Vorwissens der Lernenden möglich. Das Nutzerverhalten bzw. die am Rechner ausgeführten Aktionen können automatisch erfasst und

evaluiert werden, womit die Lernergebnisse unmittelbar in eine rechnergestützte Lernplattform übertragbar sind. Das setzt jedoch einen durchgängigen, strukturierten Entwurf von Materialien und Werkzeugen voraus [LT05].

Dieser Beitrag beschreibt ein browserbasiertes Werkzeug zur interaktiven, explorativen Visualisierung von Internetdiensten, das sowohl ergänzend zur konventionellen Präsenzlehre als auch in virtuellen Lernformen oder für das Selbstlernen eingesetzt werden kann. In Kapitel 2 werden technische Aspekte des entwickelten Frameworks dargestellt. Dessen praktischen Einsatz am Beispiel des Domain Name System (DNS) beschreibt Kapitel 3. DNS wurde aufgrund der einfachen Nachrichtenstruktur, Abläufe und Visualisierbarkeit gewählt. Abschließend werden in Kapitel 4 die bereits erschlossenen sowie derzeit noch ungenutzten Potenziale des Werkzeugs diskutiert.

2 Interaktion mit Internetdiensten

Es wurde ein generisches Framework entwickelt, dass die Interaktion mit bzw. die Visualisierung von Internetdiensten erlaubt. Über die definierten Schnittstellen ist das System um beliebige Dienste oder Visualisierungsformen erweiterbar. Die Architektur und Implementierung des Frameworks sowie die angebotenen Visualisierungen zielen durch ihre Einfachheit und Flexibilität auf eine breite Unterstützung von Lernprozessen.

2.1 Grundlegende Prinzipien

Die Erstellung interaktiver, multimedialer Komponenten ist mit einem immensen Ressourcenaufwand verbunden [Ker01]. Darum erhält die *Wiederverwendbarkeit* von bereits vorhandenen Komponenten oder auch nur Teilen davon ein großes Gewicht – nicht nur aufgrund wirtschaftlicher Erwägungen. Grundlegende technische und konzeptionelle Anforderungen wie *Plattformunabhängigkeit*, Fokussierung auf *generelle Konzepte* anstelle singulärer Objekte, *Anbindung* an bestehende Ressourcen sowie eine individuelle *Parametrisierbarkeit* und *Flexibilität* haben einen breiten Einsatz in unterschiedlichen Szenarien zum Ziel [NR02].

Weitere wesentliche Prinzipien sind ein *modularer Aufbau* aus voneinander weitgehend unabhängigen Softwarekomponenten sowie insbesondere in verteilten Systemen eine abstrahierende *mehrschichtige Architektur* [Eck95]. Diese auf definierten Schnittstellen zwischen Schichten und Objekten basierenden Techniken bewirken ebenfalls eine erhöhte Wiederverwendbarkeit, Wartbarkeit sowie Flexibilität und Erweiterbarkeit des Systems.

2.2 Architektur des Frameworks

Ausgehend von o. g. Anforderungen wurde ein generisches Framework zur Interaktion mit Internetdiensten auf Basis einer dreischichtigen Architektur entwickelt, die in Bild 1 veranschaulicht ist:

- Die *Präsentationsebene* unterstützt die Ein- und Ausgabe von dienstspezifischen Daten (Request und Response) auf Basis von diensteunabhängigen Visualisierungsmechanismen (sog. Views) in einem grafischen User Interface.

- In der *Geschäftsebene* agiert der Internetdienste-Explorer als Service Broker, der zwischen verschiedenen Diensten auf der einen und verschiedenen Visualisierungen auf der anderen Seite vermittelt.
- Auf unterster Ebene befinden sich beliebige *Internetdienste* (oder Simulatoren), die über Standardschnittstellen angesprochen werden.

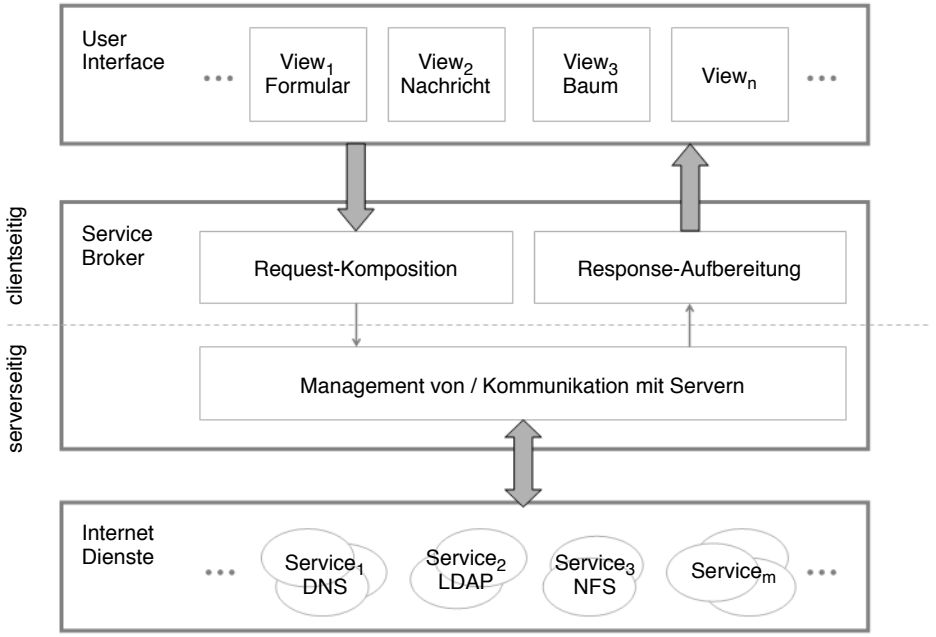

Bild 1 Dreischichtige Broker-Architektur des entwickelten Frameworks

Das Framework ist um beliebige Views oder Dienste erweiterbar, was eine insbesondere für die Gestaltung von eLearning-Prozessen unabdingbare Flexibilität bietet. Derzeit wird die Erweiterung über Factorys realisiert. Eine zweite XML-basierte Methode bietet einen einfacheren Zugang, befindet sich jedoch noch im Anfangsstadium.

Der *Service Broker* bildet das zentrale Element im Internetdienste-Explorer, da er zwischen Visualisierungen und Diensten vermittelt. Seine Komponenten sind sowohl client- als auch serverseitig aktiv. Er hält Informationen über vorhandene Server vor, die individuell erweiterbar sind, und kontaktiert entsprechend den Parametern der Anfrage (Typ, Adresse und Optionen des Dienstes) den korrespondierenden Kommunikationsendpunkt. Das Framework ist demnach als Middleware für den interaktiven Zugriff auf Internetdienste interpretierbar.

2.3 Verwendete Technologien

Als Implementierungsgrundlage für das Framework wurde aufgrund ihrer Plattformunabhängigkeit die *Java-Technologie* gewählt. Weitere Vorteile von Java bestehen zudem in der Unterstützung von verteilter Programmierung, Softwarekomponentenmodellen sowie mehrschichtigen Architekturen.

Komponentenmodelle wie *JavaBeans* [Sun97] erhöhen die Wiederverwendbarkeit von Bestandteilen des Frameworks. Komponenten mit einem geringen Kontextbezug, wie z. B. die unterschiedlichen Views, werden als in sich vollständig gekapselte Beans realisiert. Sie lassen sich somit in anderen Tools – auch außerhalb des eLearnings – auf einfache Weise erneut einsetzen.

Enterprise JavaBeans (*EJB*) [Sun03] stellen als Middleware eine flexible Schnittstelle zwischen den Komponenten des Internetdienste-Explorers bereit. Sie ermöglichen eine hinsichtlich Wartung und Wiederverwendbarkeit optimierte Implementierung der dreischichtigen Architektur.

Die Erweiterbarkeit des Frameworks lässt sich darüber hinaus durch generische Modelle für wiederkehrende Entwurfsprobleme wesentlich verbessern. Aus diesem Grund bilden *Design Patterns* [GHJ96] einen wesentlichen Eckpfeiler der vorgestellten Architektur.

3 Praxisbeispiel: Domain Name System

Das *Domain Name System* (*DNS*) realisiert die Abbildung zwischen Rechnernamen und IP-Adressen in einem hierarchischen Namensraum [Moc87a][Moc87b]. Kann ein DNS-Server die Anfrage eines Clients nicht selbst beantworten, ist deren Delegation an einen weiteren Server möglich. Basierend auf dem in Kapitel 2 eingeführten Framework wird in den folgenden Abschnitten exemplarisch der Zugang zum DNS-Dienst im Kontext eines explorativen Lernprozesses geschildert. Die Zielgruppe des Werkzeugs sind damit (neben Lehrenden bei der Vorbereitung und Durchführung von Lehrveranstaltungen) vorwiegend Studierende der Informatik im Haupt- oder Nebenfach im freien oder Vorlesungsbegleitenden Selbststudium.

3.1 Eine DNS-Anfrage

Um dem Anwender einen explorativen Zugang zum DNS zu bieten, wurde ein intuitives *User Interface* entwickelt. Bei der Erstellung einer Dienstanfrage wird von unnötig komplexen Inhalten und Darstellungen des Nachrichtenformats durch Vorbelegung bzw. Visualisierung abstrahiert. So lehnt sich die in Bild 2 gezeigte Nutzerschnittstelle in ihrer Gestaltung an den Aufbau einer DNS-Anfrage an, verbirgt jedoch irritierende Details.

Das *DNS-Protokoll* spezifiziert ein einheitliches Format für Requests und Responses. Da viele Felder nur für Antwortnachrichten benötigt werden (z. B. Recursion Available, RA), ist ihre Eingabe nicht erforderlich. Die folgenden Parameter einer Anfrage sind vom Nutzer interaktiv zu definieren:

- die Flags Opcode (Art der Anfrage) und RD (Recursion Desired) im Header
- der Anfragename in Form eines validen DNS-Namens
- der Anfragetyp zur Festlegung der Art der angefragten Information, z. B. `MX` für die Mail Exchanger oder `A` für die IPv4-Adresse einer Domain
- die Anfrageklasse zur Angabe des Adresssystems, i. d. R. `IN` für Internet-Adressen

Das User Interface unterstützt die interaktive Manipulation dieser Parameter durch Auswahllisten, Checkboxen etc. und widerspiegelt die Reihenfolge der Felder in der Nachricht. Für eine Anfrage nicht erforderliche Informationen, wie z. B. die Anzahl der

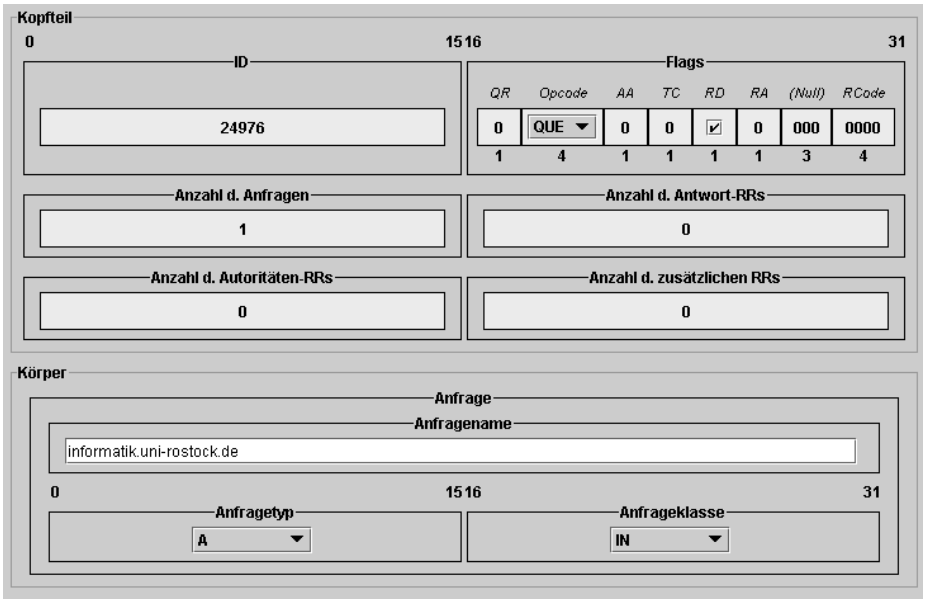

Bild 2 Grafische Nutzerschnittstelle zum DNS-Client

Antworten und Autoritäten, sind als statische Werte angezeigt. Weitere Flags und Felder werden frei generiert (z. B. der Identifikator ID) oder automatisch aus den eingegebenen Daten ermittelt (z. B. das Flag Truncated, TC). Für alle Nachrichtenfelder sind erläuternde Beschreibungen als kontextsensitive Mouse-Over-Hilfe verfügbar.

Den Ablauf einer Anfrage verdeutlicht Bild 3. Die Eingaben im Formular werden durch den Service Broker zu einer validen DNS-Anfragenachricht aggregiert. Sie wird an einen vom Nutzer ausgewählten oder frei spezifizierten DNS-Server versandt, der die Anfrage bearbeitet und einen Response zurücksendet. Für nichtrekursive Anfragen werden aus den Antworten der Server auf Basis des DNS-Delegationsmodells automatisch neue Anfragen erzeugt, so dass eine vollständige iterative Namensauflösung erfolgt. Rekursive Anfragen werden vom Anwender steuerbar entweder mit Hilfe der iterativen Vorgehensweise nachgebildet oder direkt rekursiv ausgeführt. Die Antwortnachrichten können verschieden visualisiert werden.

3.2 Nachricht oder Baum?

DNS-Antworten weisen die gleiche Struktur wie die korrespondierenden Anfragen auf, wobei die durch den DNS-Server ermittelten Informationen innerhalb von Flags (z. B. AA für eine autoritative Antwort) und Zählerfeldern (z. B. Anzahl der Antworten) im Header sowie Ressource Records mit den eigentlichen Domain-Daten im Nachrichtenkörper gekapselt sind. Aus diesen Informationen werden in einer dreistufigen *Visualisierungs-Pipeline* [HN90] aussagekräftige Darstellungen generiert:

1. Filtering zur Verfeinerung der Eingabedaten (z. B. Entfernen unnötiger Werte)
2. Mapping zur Überführung in eine geometrische Repräsentation
   (z. B. 2D-Modell eines Baumes mit Koordinaten für Knoten- und Kantenobjekte)

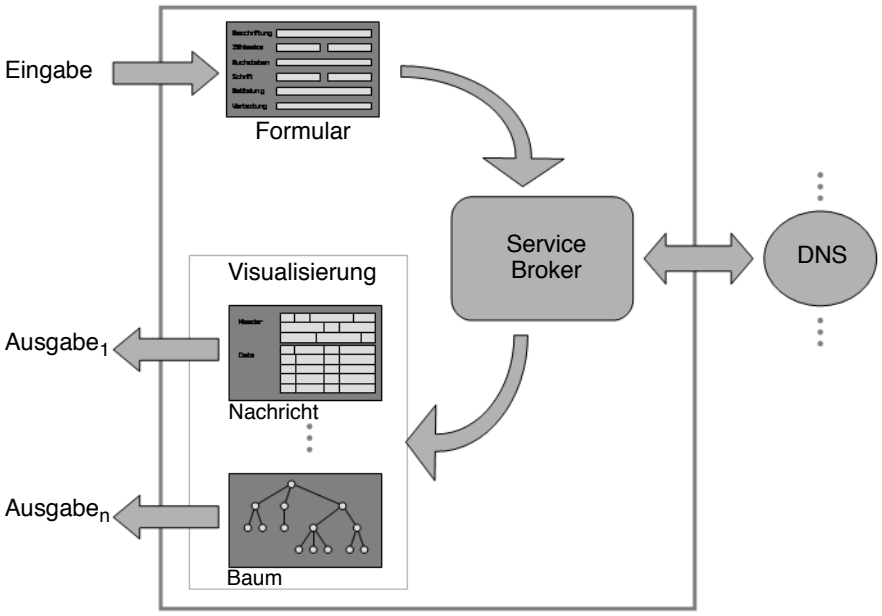

Bild 3 Datenfluss bei der Erzeugung, Bearbeitung und Visualisierung von DNS-Anfragen

3. Rendering zur Erzeugung eines sichtbaren Bildes

Da alle visuellen Repräsentationen innerhalb des Frameworks in sich abgeschlossene Beans sind, wird das zunächst nötige Filtering von einfachen Adapter-Klassen realisiert. Sie entfernen die für eine konkrete Darstellung nicht benötigten Werte aus den Nachrichten und konvertieren die verbleibenden Daten in das interne Datenformat der View. Mapping und Rendering werden dann in den View-Beans selbst berechnet. Bislang sind folgende Views verfügbar:

- textbasierte Nachrichtenansicht
- grafische Nachrichtenansicht
- Baumdarstellung

Dabei stellt die *textbasierte Nachrichtenansicht* keine Visualisierung im eigentlichen Sinne dar. Sie veranschaulicht alle in der Antwortnachricht enthaltenen Informationen, wie z. B. die Typen und Adressklassen von Resource Records sowie die enthaltenen Domain-Informationen, aber auch die korrespondierende Anfragenachricht im Detail. Durch aussagekräftige Benennungen aller Elemente (in Analogie zu Tools wie `nslookup` und `dig`) wird zudem eine abstraktere Sicht auf binär kodierte Nachrichten erreicht. So wird z. B. anstelle des Return Codes `0100` des Servers die Fehlermeldung „`Anfragetyp vom Server nicht unterstützt`“ generiert.

Die *grafische Nachrichtenvisualisierung* korrespondiert eng mit der logischen Struktur der Nachricht und ähnelt darum der formularbasierten Eingabe, wobei jedoch antwortspezifische Teile wie in Bild 4 dargestellt mit den vom Server gelieferten Informationen gefüllt sind. Durch die eindeutige visuelle Trennung und Strukturierung der Felder wird eine anschaulichere, abstraktere Sicht als in der Textdarstellung geboten.

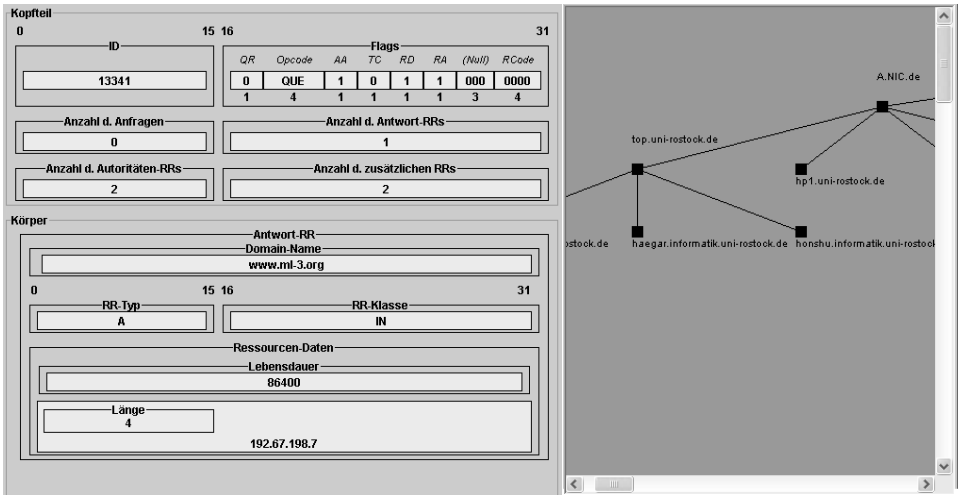

Bild 4 Gekürzte graphische Nachrichtenvisualisierung als Nachricht (links) oder Baum (rechts)

Die *Baumvisualisierung* verdeutlicht die Interaktionen des DNS-Clients mit den DNS-Servern und der DNS-Server untereinander. Wie in Bild 4 gezeigt, kann so dem Lerner das DNS-Delegationsmodell auf einfache Weise veranschaulicht werden. Die Erzeugung des Baums erfordert ein gezieltes Filtering und Mapping in der Visualisierungs-Pipeline. Beim Filtering werden zunächst die DNS-Server aus den Antworten extrahiert. Sie bilden neben dem Client die eigentlichen Knoten des Baumes, ihre DNS-Namen bzw. IP-Adressen dienen als Labels. Das Mapping erstellt daraus eine hierarchische Ordnung:

- Der DNS-Client bildet stets die Wurzel des Baumes.
- Eine *rekursive Anfrage* erzeugt einen rudimentären Baum mit nur zwei Knoten. Neben dem Client ist nur ein den Request bearbeitender (oder ggf. ablehnender) Server sichtbar; alle weiteren kontaktierten Server bleiben transparent.
- Eine *iterative Anfrage* erzeugt eine Hierarchie mit einer Ebene unter dem Client, in der die iterativ kontaktierten DNS-Server in der korrespondierenden Reihenfolge dargestellt sind.
- Eine *nachgebildete rekursive Anfrage* erzeugt i. A. einen mehrstufigen Baum. Jeder weitere kontaktierte Server ist einer neuen Ebene zugeordnet. Verzweigungen entstehen, wenn ein Server einen Fehler zurückliefert bzw. keine Rekursion unterstützt, sodass der anfragende Server iterativ vorgehen muss.

Die entstandene Datenstruktur wird durch einen geeigneten Layoutmechanismus in eine 2D-Repräsentation umgewandelt [Wal90], die im abschließenden Rendering durch Knoten-, Label- und Kantenobjekte dargestellt wird. Aufgrund der Effizienz des DNS sind die entstehenden Baumstrukturen zwar nicht sehr komplex. Es wurde dennoch ein vglw. mächtiger Layoutalgorithmus gewählt, um diese Visualisierung auch für andere Dienste einsetzen zu können.

3.3 DNS-Server oder Simulator?

Das entwickelte Framework kann nicht nur für *reale Internetdienste*, sondern auch in Verbindung mit *Simulatoren* eingesetzt werden. Aus diversen eLearning-Aktivitäten, aber auch aus Fachgebieten wie z. B. Netzwerkanalyse existieren zahlreiche Simulatoren für verschiedene netzbasierte Protokolle und Dienste. Beispielhaft seien hier der Java-basierte DNS-Simulator der University of New South Wales [Sch01] sowie die Modellierung von IP-basierten Netzen an der Universität Hamburg [SWS03] angeführt.

Da der Service Broker auf *standardisierte Schnittstellen* der Dienste zugreift, sind die angefragten Server beliebig wählbar. Zu ihrer Identifikation werden lediglich die Adresse des Zielrechners und die Portnummer des zu nutzenden Dienstes (für DNS i. d. R. Port 53) angegeben. Eine Reihe von Standardwerten sind bereits voreingestellt; sie bilden ein repräsentatives, für den Lernprozess gut geeignetes Spektrum. Es sind jedoch beliebige Adressen bzw. Ports spezifizierbar. Daher kann das Werkzeug nicht nur an eigene Server, sondern auch an beliebige Simulatoren angebunden werden (egal ob diese lokal installiert oder über das Netzwerk verfügbar sind). Die Operationsgrundlage – nicht nur für DNS, sondern auch für alle anderen Dienste – ist in jedem Fall das standardisierte Protokoll des Dienstes.

Aus pädagogischer Sicht bietet dies eine wichtige *Flexibilität*. Einerseits können die Lernenden an verschiedenen Servern deren Verhalten bei unterschiedlichen Betriebsarten studieren. Auf der anderen Seite wird es den Lehrenden ohne großen Aufwand ermöglicht, gezielt eigene Server oder Simulatoren einzubinden, deren Konfiguration jeweils auf das spezifische Lernziel ausgerichtet ist. Bei Diensten wie LDAP oder NFS ist die Auswahl dedizierter Server oder Simulatoren auch aus Sicherheitsgründen angebracht, da für Lernzwecke kein unbeschränkter Zugriff auf derartige Dienste gewährt werden sollte.

3.4 An der Uni oder zu Hause?

Einsatzbedingungen wie die Verfügbarkeit bzw. Qualität der Internetanbindung dürfen kein Ausschlusskriterium für den Einsatz eines eLearning-Werkzeugs bilden. Stets sollten geeignete Alternativen der Funktionsweise (ggf. unter Anpassung des Funktionsumfangs) entsprechend den jeweiligen Gegebenheiten zur Verfügung stehen.

Prinzipiell wird für die Durchführung einer DNS-Anfrage lediglich eine *schmalbandige Internetanbindung* benötigt, unabhängig davon ob remote ein DNS-Server oder ein Simulator kontaktiert wird. Die Paketgrößen von DNS-Request und -Response liegen in der Regel unter 512 Byte, sind also auch per Analogmodem problemlos übertragbar. Ein aufwändigerer Transfer findet allerdings für das Java-Package mit der graphischen Nutzeroberfläche und der clientseitigen Programmlogik des Internetdienste-Explorers (etwa 400 kByte) statt. Ist das Werkzeug bspw. in hypertextuelles Lernmaterial integriert, können Verzögerungen beim Download schnell zur Demotivierung und Frustration der Lernenden oder sogar zum Abbruch des Lernprozesses führen. Ohne einen breitbandigen Netzanschluss sollte das Framework daher möglichst schon im Vorfeld lokal bereitgestellt werden, ggf. über andere Distributionswege.

Arbeitet das Clientgerät *offline*, ist lediglich ein DNS-Simulator nutzbar. Entsprechend dessen Funktionsumfang sind dabei u. U. weitaus weniger Möglichkeiten gegeben als in einer realen Umgebung. Aus pädagogischer Sicht sind darüber hinaus die eingeschränkte tutorielle Betreuung und das fehlende Feedback von Lernergebnissen an eine zentrale Lernplattform zu kritisieren. Grundsätzlich ist das explorativen Erlernen von (simulierten) Internetdiensten mit dem Werkzeug jedoch auch ohne Internetanbindung möglich.

Für den typischen Einsatz im Hochschulstudium kann jedoch erfahrungsgemäß bereits von einem *breitbandigen Internetzugang* (LAN, WLAN, DSL o. ä.) auf dem Client ausgegangen werden. Daher sind ohne Probleme sowohl DNS-Server bzw. Simulatoren kontaktierbar als auch der Java-Code in angemessener Zeit übertragbar – in einem WLAN werden bei typischen Übertragungsraten unterhalb 10 MBit/s noch Transferzeiten im Millisekunden-Bereich erzielt. Die Vorteile dieses Szenarios liegen neben der Vielfalt der Dienstelandschaft vor allem in der vollständigen Kontrolle der Rahmenbedingungen durch den Lehrenden. Zudem ist eine kontinuierliche tutorielle Betreuung möglich, und das Werkzeug kann in eine übergeordnete eLearning-Umgebung integriert werden.

4 Zusammenfassung und Ausblick

In diesem Beitrag wurde ein Werkzeug für den *interaktiven Zugang zu Internetdiensten* vorgestellt, das neben der Unterstützung der Präsenzlehre insbesondere für das *explorative Lernen* geeignet ist. In einem selbstgesteuerten, spielerischen Prozess können Lernende ihr erworbenes Vorwissen zum Nachrichtenaufbau und zur Funktionsweise der Dienste festigen und vertiefen. Verschiedene Visualisierungsmöglichkeiten bieten hierfür eine wertvolle Unterstützung. Die Arbeitsweise und der Einsatz des Werkzeugs wurden am Beispiel DNS demonstriert.

Das generische Framework basiert auf Java in einer dreischichtigen Broker-Architektur. Herausragende Merkmale sind die einfache Erweiterbarkeit um beliebige Visualisierungsformen und Dienstetypen sowie die freie Kontaktaufnahme zu verschiedenen Servern oder Simulatoren. Der Service Broker wird im gängigen Hochschulszenario auf einem zentralen Server bereitgestellt und von den Clients browserbasiert über Applets kontaktiert, die z. B. in hypertextuelle Lernsysteme [WWR] eingebettet sind. Durch Verwendung von Simulatoren werden jedoch auch Stand-Alone-Systeme ohne Netzanbindung unterstützt.

Künftige *Weiterentwicklungen* werden weniger die zentralen Komponenten des Frameworks, sondern vor allem die Erweiterung der angebotenen Visualisierungen und Dienste betreffen. So ist bei den nachrichtenbasierten Ansichten z. B. die Wahl zwischen verbaler und Binär- bzw. Hexadezimalform vergleichsweise einfach realisierbar. Weiterhin sind farbliche Hervorhebungen geplant. Ferner bietet eine graphische Prozessvisualisierung für zustandsbasierte Protokolle ein hohes didaktisches Potenzial. Es ist weiterhin geplant, die Interaktion mit den Diensten künftig auch direkt in den verschiedenen Views und nicht nur in der Formularansicht zu ermöglichen. Iterative Prozesse könnten dann ergänzend durch einen Einzelschrittmodus, einstellbare Verzögerungen zwischen den Requests oder eine Abbruchmöglichkeit unterstützt werden. Möglich wäre dabei auch das freie Browsen in Hierarchien wie z. B. dem DNS-Namensraum oder einem NFS-Dateisystem. Die Analyse komplexer Hierarchien könnte dabei z. B. durch Fokus- und Kontexttechniken wie das

Hyperbolic Browsing oder raumfüllende Visualisierungstechniken wie Tree Maps unterstützt werden [CMS99]. Eine aufwändige Visualisierung steht allerdings derzeit nicht im Vordergrund der laufenden Arbeiten.

Unabhängig vom Werkzeug selbst sollte auch die Entwicklung neuer *Simulatoren* für Internetdienste vorangetrieben werden, deren Einsatz insbesondere für eingeschränkte Szenarien ohne Netzanbindung bzw. bei erhöhten Sicherheitsrisiken angebracht ist.

Darüber hinaus bietet die *Integration* einzelner Lernwerkzeuge in ein Geflecht aus digitalen Lehr- und Lernmaterialien sowie rechnergestützten Werkzeugen zu deren Produktion, Verwaltung und Nutzung ein völlig neues Potenzial für einen ganzheitlichen pädagogischen Ansatz. Wenn Lehrende bei der Konzeption von Kursen auf vielfältige interaktive Bausteine zur Unterstützung des Lehr- oder Lernprozesses zugreifen können, lässt das sinkende Produktionskosten und eine gesteigerte Qualität des Materials erwarten. Wenn den Lernenden eine konsistente und möglichst homogene Umgebung für alle mit dem Lernprozess verbundenen Aktivitäten angeboten wird, können organisatorische oder technische Hemmnisse minimiert werden. Wenn die transparente, systemübergreifende Kontrolle und Dokumentation individueller Lernprozesse durch zentrale Lernplattformen möglich ist, erlaubt die schnellere Rückkopplung zum Lehrenden eine Optimierung des Lernens. Diese Vision geht weit über die Entwicklung einzelner Lernwerkzeugs hinaus; sie zielt vielmehr auf die Verwirklichung einer durchgängigen Lehr- und Lernumgebung.

5 Literatur

[CMS99] S. K. Card, J. D. Mackinlay, B. Shniderman (Ed.s): „Readings in Information Visualization: Using Vision to Think“. Morgan Kaufmann Publishers, San Diego/USA 1999.

[Coo98] J. W. Cooper: „The Design Patterns Java Companion“. IBM, 1998. http://www.patterndepot.com/put/8/JavaPatterns.htm (*besucht im April 2005*)

[Eck95] W. W. Eckerson: „Three-Tier Client/Server Architecture: Achieving Scalability, Performance, and Efficiency in Client Server Applications“. Open Information Systems, Vol. 10, 1995.

[FHW04] K. Fiolka, K. Heidtmann, B. Wolfinger: „Experimentelles und exploratives Lernen mit selbstentwickelten eLearning Werkzeugen im Bereich der Telematik“. Die 2. e-Learning Fachtagung Informatik (DeLFI), Paderborn 2004.

[GHJ96] E. Gamma, R. Helm, R. Johnson, J. Vlissides: „Entwurfsmuster: Elemente wiederverwendbarer objektorientierter Software“. Addison Wesley, Bonn 1996.

[HN90] R. B. Haber, D. A. McNabb, „Visualization Idioms: A Conceptual Model for Scientific Visualization Systems”. IEEE Visualization in Scientific Computing, 1990.

[KKK02] W. Kalfa, R. Kröger, F. Köhler: „Integrating Simulators and Real-Life Experiments into an XML-based Teaching and Learning Platform“. World Conference on Educational Multimedia, Hypermedia, and Telecommunications (Ed-Media), Denver/Colorado/USA 2002.

[Ker01] M. Kerres: „Multimediale und telemediale Lernumgebungen“. Oldenbourg Verlag, 2001

[LT05] U. Lucke, D. Tavangarian: „Strukturierte Lehr- und Lernmodelle für die Ausbildung im Fachgebiet Technische Informatik“. Information Technology (it), Vol. 47/3, 2005.

[Med] Universitätsklinikum Heidelberg: Projekt "medicase: CAMPUS-Pädiatrie". http://www.medicase.de/ (*besucht im April 2005*)

[Moc87a] P. Mockapetris: „Domain Names – Concepts and Facilities“. IETF Network Working Group, RFC 1034.

[Moc87b] P. Mockapetris: „Domain Names – Implementation and Specification“. IETF Network Working Group, RFC 1035.

[NR02] Th. Naps, G. Rößling et al.: „Exploring the Role of Visualization and Engagement in Computer Science Education“. Report of the ACM Working Group on Innovation and technology in computer science education, 2002.

[NR03] Th. Naps, G. Rößling et al.: „Evaluating the Educational Impact of Visualization“. Report of the ACM Working Group on Innovation and technology in computer science education, 2003.

[Ple04] J. Pleumann: „Erfahrungen mit dem multimedialen, didaktischen Modellierungswerkzeug DAVE“. Die 2. e-Learning Fachtagung Informatik (DeLFI), Paderborn 2004.

[Sch01] G. Schokmann: „DNS Simulator“. Projektarbeit, University of New South Wales/Australia, 2001.

[Sch03] S. J. Schmidt (Hrsg.): "Der Diskurs des Radikalen Konstruktivismus", Suhrkamp Verlag, 2003.

[Ste00] V. Steiner: "Exploratives Lernen", Pendo Verlag, 2000.

[Sun97] Sun Microsystems: „JavaBeans™ API Specification, v 1.01-A“, 1997. http://java.sun.com/products/javabeans/ (*besucht im April 2005*)

[Sun03] Sun Microsystems: „Enterprise JavaBeans™ Specification, v 2.1“, 2003. http://java.sun.com/products/ejb/ (*besucht im April 2005*)

[SWS03] C. Scherpe, B.E. Wolfinger, I. Salzmann: „Model Based Network Emulation to Study the Behavior and Quality of Real-Time Applications“. 7th IEEE Int. Symp. on Distributed Simulation and Real Time Applications (DS-RT 2003), Delft/Niederlande 2003.

[Wal90] J. Q. Walker, II.: „A node-positioning algorithm for general trees“. Software – Practice & Experience, Vol. 20, Issue 7, John Wiley & Sons, Inc., New York, 1990.

[WWR] Universität Rostock: Projekt „Wissenswerkstatt Rechensysteme“. http://www.wwr-project.de/ (*besucht im Juli 2005*)