Scone: Ein Framework zur evaluativen Realisierung von Erweiterungen des Webs

Harald Weinreich, Volkert Buchmann, Winfried Lamersdorf

Arbeitsbereich Verteilte Systeme und Informationssysteme (VSIS)
Fachbereich Informatik, Universität Hamburg
Vogt-Kölln-Str. 30, 22527 Hamburg

{weinreich|lamersdorf}@informatik.uni-hamburg.de
volkert@spacecactus.net

Zusammenfassung: Projekte, die sich mit der Entwicklung und Evaluation von Konzepten, Techniken und Werkzeugen zur Verbesserung der Benutzbarkeit des Webs befassen, stehen oft vor dem Problem, dass die Implementation entsprechender Prototypen sehr aufwendig ist. Es stand bisher kein angemessenes Framework zur Verfügung, das solche Entwicklungen vereinfachen könnte. Dieses Paper stellt das Framework „Scone“ vor, das die prototypische Entwicklung und Evaluation von unterschiedlichen Arten von Web-Erweiterungen – insbesondere solcher zur Navigation und Orientierung im Web – erleichtert. Das Framework bietet eine Reihe von Komponenten, welche es u.a. erlauben, die Darstellung der Dokumente im Browser zu ändern, auf Benutzeraktionen mit dem Browser zu reagieren, den Browser zu steuern und auch selbsttätig Informationen aus dem Netz zu sammeln. Zusätzlich wird die Evaluation solcher Systeme durch Benutzbarkeitsstudien unterstützt.

1 Die Herausforderungen der Navigation im Web

Das „Surfen“ im Web stellt auf den ersten Blick keine großen Anforderungen an die Benutzer. Die notwendigen Bedienkonzepte sind vergleichsweise schnell und leicht zu erlernen. Bei genauerer Betrachtung kann aber eine ganze Reihe von gravierenden Benutzbarkeitsproblemen festgestellt werden. Diese werden nicht nur durch wenig fachgemäß entworfene und realisierte Web-Sites verursacht, sondern sind vielmehr größtenteils systemimmanent. Einige Schwierigkeiten für Benutzer sind in der *Hypertext-Charakteristik* des Webs begründet, andere werden durch die Mängel der inzwischen leider etablierten und daher schwer zu ändernden *User-Interfaces der Browser* hervorgerufen. Weitere Probleme verursachen die teilweise *unzureichenden Sprachen und Protokolle* des Webs, sowie sein globaler und strukturloser *verteilter Aufbau*.

Das Web erfüllt durch die in den Dokumenten enthaltenen Verweise auf andere Objekte das Schlüsselkriterium für *Hypertext-Systeme.* Dies ist wohl zugleich seine größte Stärke als auch Ursache für viele Herausforderungen bei seiner Benutzung. Conklin charakterisierte bereits in [10] die gravierendsten Probleme bei der Navigation im *Hyperspace*:

- *Disorientation* – Benutzer verlieren leicht die Orientierung; es gibt keine globalen “Wegweiser” oder “Landkarten”. Die Strukturen des Hyperspace sind zudem viel

zu kompliziert um sie zu überschauen oder auswendig zu lernen. Dies führt zu dem so genannten “Lost in Hyperspace”-Problem.

- *Cognitive Overhead* – Um im Hyperspace zum Ziel zu kommen, müssen Benutzer neben den Herausforderungen der Navigation und Orientierung auch ihre Aufgabe im Auge behalten. Daraus ergibt sich eine doppelte Belastung des Kurzzeitgedächtnisses.

Diese bereits vor der Einführung des World Wide Webs erkannten Probleme sind auch für das Web relevant; dennoch berücksichtigen die im Web benutzten Konzepte sie kaum und greifen bewährte Lösungen älterer Hypertextsysteme nicht auf.

Ebenso sind einige gravierende Schwachstellen des Webs in den *User Interfaces heutiger Browser* auszumachen. Sie bieten von sich aus vergleichsweise wenige und unzureichende Werkzeuge, um die Navigation und Orientierung im Web zu erleichtern [4]. Während einerseits die technischen Möglichkeiten der Browser ständig erweitert werden, beispielsweise im Bereich der XML-Sprachfamilie oder bei Programmiersprachen wie ECMA-Script[1], hat sich andererseits bei den Benutzungsschnittstellen seit den ersten Versionen von Mosaic[2] kaum etwas verändert. Andere Hypertext-Systeme leisteten bereits vor langem mehr. Beispielsweise bot der Browser Harmony des Systems Hyper-G grundsätzlich eine Übersicht zur Site-Struktur und Zugriff auf ein integriertes Suchsystem [2]. Bei Web-Browsern führt hingegen schon der Back-Button wegen seines Stack-basierten Charakters zu Problemen [9]. Ebenso ist die History (auch „Verlauf“ genannt), die alle in den letzten Wochen besuchten Seiten aufzeigt, immer noch wenig ausgereift. Sie macht das Wiederfinden von früher besuchten Dokumenten aufgrund ihrer schlechten Repräsentation schwierig und wird folglich selten genutzt [24]. Ein drittes Beispiel sind „Bookmarks“ (bzw. „Favoriten“), deren Verwaltung vielen Benutzern als zu aufwändig erscheint [1; 16].

Die Schwächen von *HTML und HTTP* wurden bereits in vielen Bereichen angegangen, dennoch sind noch viele Probleme ungelöst. Beispielsweise führen die ins Dokument eingebetteten und semantisch ausdrucksschwachen Links zur so genannten “*Hub and Spoke*”-Navigation, bei der ein Benutzer nacheinander mehreren Verweisen einer Seite folgt, um herauszufinden, welche Informationen sich hinter ihnen verbergen, bis er letztendlich das gewünschte Dokument gefunden hat [24]. Hinzu kommt die mangelhafte Einbindung von semantischen oder Meta-Informationen in HTML-Dokumente, wodurch die Effizienz und Erstellung von Katalogsystemen und integrierenden Diensten erheblich erschwert werden. Diese Schwächen werden durch die *globale Verteilung* der Informationen und Dienste des Webs auf Millionen von Servern noch verstärkt. Das WWW kann als eine sich stets verändernde, inkonsistente Datenbasis angesehen werden, die zudem so umfangreich und dabei ohne einheitliche Struktur ist, dass heutige Suchsysteme auf eine Anfrage in der Regel zu viele ungewünschte oder unpassende Treffer ausgeben. Doch auch für die Betreiber einer Web-Site ist es schwer, die eigenen Daten konsistent zu halten. Innerhalb einer Web-Site können Content-Management-Systeme für konsistente Links sorgen, fehlerhafte externe Verweise lassen sich jedoch mangels standardisierter Konzepte und *Protokolle*[3]

[1] ECMA-Script ist ein internationaler Standard für Scriptsprachen im Web und damit der Versuch, Inkompatibilitäten der Ausgangssprache JavaScript zu beseitigen.

[2] Mosaic war der erste Web-Browser mit GUI für X-Windows-Systeme. Mehr Informationen unter http://archive.ncsa.uiuc.edu/SDG/Software/Mosaic/

[3] Es gab hierzu bereits einige Lösungsansätze. Ein Beispiel ist der Atlas Link-Server [22].

bis heute nicht immer vermeiden. So konfrontieren „Broken Links“ die Benutzer immer wieder mit wenig hilfreichen Fehlermeldungen. Untersuchungen zeigen, dass im Herbst 2002 durchschnittlich über 5% der Links fehlerhaft sind [17].

Zusammengefasst sind die Simplizität der Grundkonzepte des Webs und die minimalistischen User Interfaces der Browser die größten Schwachpunkte des Webs: Die verwendeten Sprachen und Protokolle werden vielen Problemen von verteilten Hypertextsystemen nur unzureichend gerecht. Konzepte zur Behebung vieler altbekannter Hürden bei der Benutzung verteilter Hypertextsysteme, Dienste und Informationssysteme fehlen oder ließen sich bis heute nicht umsetzen. Zahlreiche Schwierigkeiten wären absehbar und vermeidbar gewesen, wenn für diese Konzepte zuvor eine Entwicklung von Prototypen und deren Evaluation stattgefunden hätte.

2 Strategien für ein besser benutzbares Web

Viele Forschungsprojekte versuchen, die Benutzbarkeit des Webs zu verbessern. Um dies zu erreichen, wird zum einen versucht, auf Basis der existierenden Technologien und Infrastrukturen bessere Interfaces für Benutzer zu entwickeln. Andere Projekte arbeiten daran, die technischen Grundlagen des Webs zu erweitern. Um den tatsächlichen Nutzen solcher Konzepte und Werkzeuge beurteilen zu können, müssen entsprechende Prototypen erstellt und in Anwendungsszenarien evaluiert werden.

Die Ziele von Projekten, die sich mit einer Verbesserung der Benutzbarkeit des Webs beschäftigen, können in der Regel den drei Kernbereichen Orientierung, Navigation und Kollaboration zugeordnet werden.

Um eine einfache *Orientierung* im Web zu gewährleisten, müssen die Fragen “Wo war ich?”, “Wo bin ich?” und “Wo kann ich hin?” jederzeit klar beantwortet werden können. Orientierungshilfen können beispielsweise innerhalb des Dokuments, der Site, dem thematischen Kontext oder der topologischen Umgebung geboten werden. Ein Ansatz in dieser Richtung sind graphische Übersichten [11].

Zudem soll zielgerichtete *Navigation* möglich sein. Neue Informationen müssen problemlos gefunden und bereits besuchte Dokumente einfach wieder erreicht werden können. Ein Beispiel einer Erweiterung, die die Navigation im Web vereinfacht, ist der „Google Toolbar“[4]. Er positioniert sich unter der Adressleiste des Internet Explorers und erlaubt mittels neuer Interaktionselemente einen direkten Zugriff auf viele der Funktionen von Google; es lassen sich beispielsweise direkt Suchanfragen stellen oder verwandte Seiten zum aktuellen Dokument ermitteln.

Die *Kollaboration* im Web wird bis heute kaum unterstützt. Antworten auf Fragen wie „Welche Dokumente können mir Benutzer mit ähnlichen Zielen empfehlen?“ und „Wie hilfreich fanden andere Benutzer diese Seite?” enthalten wertvolle Hinweise für die Suche nach Informationen. Sowohl durch die synchrone als auch durch die asynchrone Zusammenarbeit können Benutzer von den Erfahrungen anderer profitieren [23]. Eine Reihe solcher Anwendungen finden sich beispielsweise im E-Commerce-Umfeld, jedoch handelt es sich dabei nicht um systemimmanente, server-übergreifende Dienste des Webs.

[4] Den Google Toolbar gibt es kostenlos für den Internet Explorer: http://toolbar.google.com/

XLink und die *Semantic Web Initiative* sind Beispiele für Bestrebungen, die mittels Erweiterungen auf technischer Ebene den Umgang mit dem Web vereinfachen und erweitern sollen. XLink ist im Zusammenhang mit XML als Ersatz für die simplen eingebetteten Links von HTML vorgesehen. Beispiele für die erweiterten Navigationsmöglichkeiten von XLink sind bidirektionale Verweise und Verweise mit multiplen Zielen. Des weiteren erlaubt XLink das Speichern von Links außerhalb der Dokumente in so genannten Link-Datenbanken. XLinks können mit semantischen Informationen versehen werden, wodurch beispielsweise die benutzerspezifische Filterung von Verweisen möglich wird [12]. Bisher stehen den XLink-Techniken allerdings nur vage Konzepte für geeignete User Interfaces gegenüber – diese wurden bewusst außen vor gelassen. Probleme bei der Umsetzung werden in [27] beschrieben.

Die Semantic Web Initiative geht einen anderen Weg. Es wird hierbei unter anderem versucht, das Web mittels der Sprache RDF um Meta-Informationen anzureichern und so für Maschinen lesbar zu machen. Dies soll Systeme ermöglichen, die für den Benutzer Informationen zusammensuchen und kombinieren [3]. Die Beschreibungen der User Interfaces hierzu sind jedoch vergleichsweise unbestimmt, und praxisreife technische Realisierungen fehlen. Zu bedenken ist zudem, dass Untersuchungen auf Autorenprobleme mit dem Hinzufügen von Meta-Informationen zu Dokumenten und Verweisen hingewiesen haben [25].

Die Schwächen heutiger User Interfaces und die Probleme bei der Umsetzung neuer technischer Standards des Webs machen deutlich, dass die prototypische Evaluation der technischen Konzepte in den gegebenen Anwendungsszenarien ein integraler Bestandteil der Entwicklung sein sollte. Für viele der technischen Erweiterungen fehlen bisher ausgereifte ergonomische Realisierungen, so auch bei XLink und RDF. Da sich diese Konzepte letztendlich daran messen lassen müssen, ob sie eine echte Verbesserung und eine Vereinfachung für die Benutzer darstellen, sollten sie schon vor ihrem Einsatz prototypisch in Anwendungsszenarien evaluiert und gegebenenfalls angepasst werden [21]. Diese Problematik war ein ausschlaggebender Motivationsfaktor, um ein Framework zu erstellen, das die einfache Erstellung von Prototypen für möglichst viele Konzepte in diesem Themenbereich ermöglicht. Zusätzlich soll die Evaluation der Prototypen unterstützt werden, damit die gewonnenen Daten wieder in den Entwicklungsprozess einfließen und so zu einem qualitativ hochwertigen Ergebnis beitragen können.

3 Anforderungen an das Framework

Die vorherigen Betrachtungen sind Ausgangspunkt für die Konzeption und Umsetzung des Frameworks *Scone*. Im Vordergrund stehen dabei die Flexibilität des Frameworks und die Unterstützung aktueller Sprachen und Technologien. So soll das Framework sowohl Möglichkeiten bieten, schnell experimentelle User Interfaces für neue Basistechnologien zu realisieren, als auch neue Konzepte für die Unterstützung der Orientierung, Navigation und Kollaboration im Web umzusetzen.

Bei der Entwicklung des Frameworks war zu bedenken, dass es recht unterschiedliche Kategorien von Konzepten zur Erweiterung der User Interfaces von Web-Browsern gibt [5; 8]. Einige Konzepte versuchen die *Darstellung der Dokumente im*

Browser zu verändern, andere fügen *dem Browser selbst neue Interface-Elemente* hinzu. Des weiteren gibt es Erweiterungen, die ein *eigenes Fenster* in Ergänzung zum Browser anbieten, oder gar *unabhängig vom Browser* funktionieren, um ihn für bestimmte Aufgaben zu ersetzen.

Auch die Personalisierbarkeit ist je nach Ansatz unterschiedlich ausgeprägt. Einige Konzepte verzichten auf die Unterscheidung von Benutzern und arbeiten aufgaben- oder sitzungsbasiert. Oft ist eine Differenzierung aber notwendig, damit Dienste personalisiert werden können oder Benutzer die Möglichkeit zur Kollaboration erhalten.

Zudem kann die Art der zur verbesserten Orientierung und Navigation zusätzlich benötigten Informationen stark variieren. Einige Werkzeuge benötigen zusätzliche Daten aus dem Netz, andere analysieren oder adaptieren die Informationen, die der Benutzer aus dem Netz abruft. Entsprechend vielseitige Schnittstellen sollten für die Entwickler von neuen Web-Erweiterungen zur Verfügung stehen: so muss das Framework beispielsweise Proxy-, Server- und Robot-Funktionalität bereitstellen.

Diesen unterschiedlichen Anforderungen kann ein Framework am besten mit einem modularen Aufbau gerecht werden; einfach kombinierbare, programmierbare und erweiterbare Komponenten können so die jeweiligen Ansätze abdecken. Zudem fördert eine komponentenorientierte Architektur das Einbinden von anderen Toolkits und Bibliotheken, etwa zur Bearbeitung von XLinks oder RDF.

4 Die Architektur des Scone-Frameworks

Scone ist als objektorientiertes Framework [13, S.26] in Java implementiert worden. Es bietet eine Reihe von Klassen und Interfaces, die je nach nötiger Flexibilität zumeist durch Komposition, zum Teil aber auch durch Subclassing Einsatz finden. Die Klassen und Interfaces des Framework können in vier Kernkomponenten und eine Reihe von Hilfskomponenten zusammengefasst werden (s. Abb. 1).

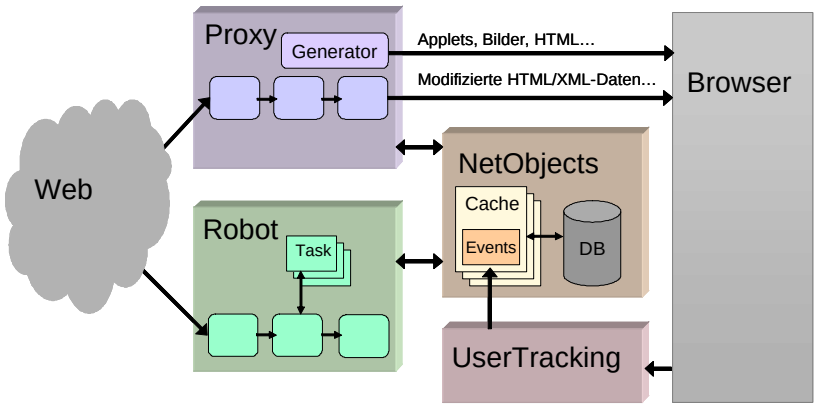

Abbildung 1: Die Kernkomponenten von Scone

Die erste der vier Basiskomponenten ist die Realisierung eines objektorientierten Datenmodells, das die wichtigsten Basiskonzepte des Internets wie URLs, Dokumente und Links als persistierbare Java-Objekte, die *NetObjects*, bereitstellt. Eine weitere

Komponente ist ein programmierbarer *Proxy*, der die zwischen Server und Browser transferierten Daten analysieren und modifizieren kann. Ein einfach zu programmierender, flexibler und nebenläufiger *Robot* ermöglicht das aktive Laden von Informationen aus dem WWW. Die vierte Kernkomponente ist das *UserTracking*, welches es erlaubt, die Aktionen unterschiedlicher Benutzer mit dem Browser zu verfolgen und aufzuzeichnen. Diese vier Komponenten werden von einer Reihe von Hilfskomponenten unterstützt. Die Komponenten sind in eine Architektur eingebunden, die es mittels eines Plugin-Konzepts erlaubt, sie zu einer Vielzahl von Erweiterungen für das Web zu kombinieren.

Die NetObjects

Die *NetObject*-Komponente stellt Möglichkeiten zur Verfügung, die in Bezug auf das WWW elementaren Objekte zu repräsentieren und bei Bedarf auch zu persistieren. Dazu gehören beliebige, durch URLs identifizierte Objekte des Internets, insbesondere aber auch Web-Dokumente mit Meta-Daten und verschiedene topologische Informationen, beispielsweise Links. Zudem lassen sich die Benutzer und ihre Aktionen darstellen. Dieses Modell lässt sich leicht anpassen und erweitern. Da das Speichern auf Sekundärspeicher vergleichsweise langsam ist und Techniken wie CASTOR oder EJBs beim Persistieren von Daten recht schwerfällig sind oder umfangreiche Application-Server voraussetzen, wurde für Scone ein optimiertes, leichtgewichtiges Caching-Konzept entwickelt, dass ein performantes und einfach adaptierbares Mapping von Java-Objekten auf relationale Datenbanken ermöglicht. Scone benutzt das Datenbanksystem MySQL, da es eine außergewöhnlich hohe Performanz aufweist. Es gibt zudem eine weitere Implementation, bei der eine Anbindung an die objektorientierte Datenbank Poet FastObjects t7 umgesetzt wurde[5].

Der Proxy

Eine weitere Komponente ist der programmierbare *HTTP Proxy*. Er wird benutzt, um die zwischen Server und Browser transferierten Daten zu analysieren und zu modifizieren. Scone verwendet und erweitert hierfür das WBI Development Kit[6] vom IBM Almaden Research Center [18]. Scone-Plugins können sich mit Hilfe des Proxys in den Datenstrom zwischen Browser und Server einklinken und haben dadurch Zugriff auf die übertragenen Web-Dokumente. Auf diese Weise kann das im Browser dargestellte Dokument beliebig angepasst und ergänzt werden. WBI wurde für Scone um einige Funktionen erweitert. So wurde zur Effizienzsteigerung die ursprüngliche Byte-basierte Verarbeitung der Web-Dokumente durch eine Token-basierte Verarbeitung ergänzt, wobei ein Token beispielsweise einen HTML-Tag oder einen Text repräsentiert. Die Dokumente werden automatisch in verschiedene Token zerlegt und in dieser Form den Plugins angeboten. Im Zuge einer Kooperation mit IBM wurde das Token-basierte Verarbeitungskonzept in WBI integriert. Durch die Transcoding-Fähigkeiten von WBI ist es Scone möglich, die Transformation und Adaption von unterschiedlichen Datenformaten während der Übertragung vorzunehmen [15]. Hierdurch können beispielsweise User Interfaces für neue technische Konzepte wie XLink erprobt werden.

---

[5] Mehr Informationen zu Poet FastObjects t7 unter: http://www.fastobjects.de/

[6] Das WBI-DK ist für den nicht-kommerziellen Einsatz kostenlos erhältlich unter: http://www.almaden.ibm.com/cs/wbi/

Der Robot
Die dritte Basiskomponente ist ein *Robot*. Er ist ähnlich einem Suchmaschinen-Robot in der Lage, ganze Web-Sites zu indizieren, kann darüber hinaus aber flexibel programmiert werden und wurde für den Einsatz auf Client-Seite optimiert. So kann der Robot je nach Anforderung nebenläufig mehrere Seiten gleichzeitig übertragen, oder auch lastminimierend eingesetzt werden und auf lokale (Netz-)Inaktivität warten. Er stellt ein *Classifier-Filter*-Konzept zur Verfügung, mit dessen Hilfe sowohl Links als auch Dokumente erst aufgrund ihrer Eigenschaften klassifiziert werden und dann entsprechend den Filterbedingungen behandelt werden. Mittels dieses Konzeptes lässt sich unter anderem angeben, welche Links weiterverfolgt und wie mit den Ziel-Dokumenten verfahren werden soll. Der Robot kann dann beispielsweise den Dokumenttyp ermitteln, die Antwortzeiten eines Servers abschätzen oder Titel und Länge eines Dokuments bestimmen. So kann der Scone Robot als ein User Agent agieren, der für den Benutzer hilfreiche Informationen sammelt.

Das UserTracking
Das *UserTracking* erlaubt es, Benutzeraktionen zu verfolgen und zu protokollieren. Es werden Interaktionen des Benutzers mit dem Browser, wie zum Beispiel der Klick auf einen Link, das Drücken des Back-Buttons oder das Ausfüllen eines Formulars berücksichtigt. Solche Aktionen erzeugen unterschiedliche Java-Events, über die ein Scone-Plugin – z.B. eine Browser-Erweiterung – mittels des Observer-Patterns informiert werden kann. Benutzer können bei Bedarf unterschieden werden; hierfür stellt Scone grundlegende Verwaltungsfunktionen wie die Registrierung und das Login zur Verfügung. Scone bietet drei unterschiedliche Varianten des UserTrackings an, die sich in der Detailliertheit der protokollierten Daten unterscheiden. Die einfachste Variante ermittelt mit Hilfe von eingebettetem JavaScript-Code, welche Dokumente wie lange in den Browser-Fenstern dargestellt wurden, nach welcher Zeit der Benutzer eine Seite wieder verlassen hat und welcher Link angeklickt wurde. Die zweite Variante bindet zudem ein unsichtbares Applet in die Seite ein, mittels dessen sich auch beliebige Aktionen auf der Seite (z.B. Tastatureingaben und Mausbewegungen) protokollieren lassen. Die komplexeste Variante verwendet zusätzlich ein Hilfsprogramm in C, das sich direkt in den Internet Explorer integriert und so genau ermitteln kann, welche Bedienelemente im Browser angeklickt wurden.

Hilfskomponenten
Neben diesen vier Basiskomponenten stehen eine Reihe weiterer Hilfskomponenten innerhalb der Scone-Architektur zur Verfügung:

- Scone bietet für die Verwaltung der Plugins ein GUI, das es erlaubt, Plugins zu registrieren und zu konfigurieren. Zur Speicherung der Konfigurationsparameter einzelner Plugins wurde ein XML-Schema definiert, aus dem automatisch ein grafisches Java-Interface zum Modifizieren der Werte generiert wird.
- Zur Analyse von HTML-Dokumenten steht ein *Standard Document Parser* zur Verfügung. Er extrahiert unterschiedliche Meta-Informationen wie die Sprache, Schlüsselwörter oder die Anzahl der Links. Zudem erzeugt er mittels heuristischer Methoden eine Kurzzusammenfassung.
- Eine wichtige Hilfskomponente ist die Browsersteuerung. Die Kommunikation mit dem Browser geschieht zum einen über ein durch den Proxy eingebundenes Applet. So lassen sich beispielsweise weitere Browser-Fenster öffnen, das ange-

zeigte Dokument oder die Position und Größe des Fensters ändern. Einige in C geschriebene Hilfsprogramme erlauben weitere Funktionen, wie das Starten eines neuen Browsers oder das Löschen der Browser-History.
- Eine beliebte Möglichkeit zur Repräsentation von Web-Seiten stellen Thumbnails dar. Scone kann auf Java basierende Komponenten zum Rendern von HTML einbinden[7] und so Web-Dokumente als Grafiken zur Verfügung stellen.
- Das Scone Framework wird durch ein Paket zur Evaluation der entwickelten Prototypen abgerundet. Mit Hilfe dieser Komponente lassen sich einfach GUIs erstellen, die Benutzer in ein Evaluationsszenario einführen und sie mittels interaktiver Fragebögen durch Aufgaben leiten. Die Eingaben und Aktionen der Benutzer können zur späteren Auswertung abgespeichert werden.

Durch die kombinierte Nutzung dieser Komponenten können vergleichsweise schnell und einfach Navigationswerkzeuge unterschiedlichster Art erstellt werden. Dabei werden von den meisten Komponenten je nach Anforderung unterschiedlich hohe Abstraktionsebenen angeboten. Ein Plugin kann beispielsweise mittels des Proxys nicht nur in den Bytestrom vom und zum Browser eingreifen, HTML-Dokumente können auch als eine Folge von Objekten wie HTML-Tags, Links und Text bearbeitet werden. Eine noch höhere Abstraktionsebene wird durch die NetObjects-Komponente erschlossen. Sie stellt Meta-Informationen des Dokuments wie Sprache, Anzahl der Links und eine Zusammenfassung bereit.

5 Möglichkeiten und Grenzen von Scone

Scone wurde als ein Framework entwickelt, das die Realisierung von möglichst vielen Ideen zur besseren Benutzbarkeit des Webs ermöglichen sollte. Dabei gibt es Konzepte, die sich mit Scone schnell und einfach realisieren lassen und andere, bei denen Scone nicht sinnvoll einsetzbar ist.

Scone eignet sich besonders gut, um Erweiterungen an bestehenden Systemen vorzunehmen. So können serverseitige Werkzeuge HTML-Dokumente mit zusätzlichen Navigationselementen anreichern oder neuartige Serverdienste erproben. Werkzeuge auf der Seite des Clients können beispielsweise innovative Navigationsmittel in einem Fenster neben dem Browser darstellen und dabei mit dem Browser interagieren oder auch neue benutzerbezogene Orientierungshilfen in Webseiten einbringen. Auch der Einsatz in Workgroups, um eine breite Evaluationsbasis zu erhalten, oder um Kollaborationskonzepte zu realisieren, ist dank des verwendeten Proxys möglich.

Eine nahtlose Integration in die Bedienelemente des Browsers, wie beim Google Toolbar, ist mit Scone momentan allerdings noch nicht umzusetzen. Dies liegt zum Teil an der eingesetzten Sprache Java, die nicht den Zugriff auf alle System- und Browserfunktionen ermöglicht, dafür aber den Vorteil der Plattformunabhängigkeit bietet. Allerdings gibt es vielversprechende erste Ergebnisse mit dem Browser Mozilla, der eine beliebige Gestaltung aller User-Interface-Elemente mittels der Sprache XUL zulässt. Auch auf Serverseite sind Grenzen erkennbar. Zwar bietet Scone die Möglichkeit, als erweiterte Kopie eines beliebigen Servers eingesetzt zu werden,

[7] Beispielsweise kann die *WebWindow-Komponente* von Horst Heistermann eingesetzt werden. Eine Testversion ist unter http://home.earthlink.net/~hheister/ erhältlich.

aber eine Integration, z.B. in Application Server, ist bisher nicht vorgesehen. Bei unseren bisherigen Projekten waren diese Einschränkungen allerdings nicht gravierend, da sich entweder eine gute Annäherung an die optimale Lösung finden ließ oder sie mit kleinen Erweiterungen des Frameworks durch Hilfsprogramme in C gelöst werden konnten.

Strukturelle Probleme bieten Objekte in Binärformaten wie Flash, die bisher von Scone nicht verarbeitet werden können. Ferner lassen sich mittels HTTPS verschlüsselt übertragene Objekte nicht von dem Proxy analysieren oder verändern.

Durch die fehlende direkte Einbindung von Scone in bestehende Browser und Server und die benötigten Komponenten wie WBI und MySQL ist nicht zu erwarten, dass Endprodukte mit Scone gefertigt werden. Die Stärke von Scone liegt in der Möglichkeit, schnell experimentelle Erweiterungen für bestehende Systeme zu implementieren um diese dann zu evaluieren. In kurzer Zeit können so Browser, bestimmte Web-Seiten oder ganze Sites um neue Techniken oder User Interfaces ergänzt werden, ohne die komplizierten und teilweise proprietären APIs heutiger Browser oder Server verwenden zu müssen. Diese bieten zudem zumeist nicht die Möglichkeiten und Werkzeuge von Scone, um neue Hilfsmittel zur Navigation und Orientierung zu realisieren. Und nicht zuletzt lässt sich Scone in *einer* API für Erweiterungen auf Client- und Serverseite einsetzen.

6 Mit Scone entwickelte Prototypen

Es wurde bereits eine ganze Reihe von unterschiedlichen Plugins für Scone realisiert. Die interessantesten sollen hier kurz vorgestellt werden.

Die Browsererweiterung **HyperScout** [26] ist ein Scone-Plugin, das die im Browser dargestellten Dokumente modifiziert. Alle HTML-Dokumente werden bei der Übertragung durch den Scone-Proxy um zusätzliche Funktionen ergänzt, die es erlauben, zu allen Links einer Web-Seite automatisch generierte, ergänzende Informationen in *Popups* darzustellen. Diese Popups erscheinen, sobald der Benutzer die Maus etwas längere Zeit über einen Link hält. So können Benutzern bei Bedarf schon vor Auswahl eines Links mehr Informationen zum Ziel angeboten werden: *Ist das Dokument verfügbar? Welchen Inhalt hat es? Wann war ich bereits auf der Seite?* Diese zusätzliche Funktionalität wurde ermöglicht, indem HyperScout alle Link-Elemente einer Seite um JavaScript-Events erweitert, die mittels LiveConnect ein ebenfalls eingebundenes Java Applet ansteuern. Dieses Applet stellt bei Bedarf ein Popup neben dem Mauszeiger dar. Die angezeigten Informationen werden unter anderem durch den Robot gesammelt und vom Applet bei Bedarf direkt von Scone geladen. HyperScout wurde bereits mehreren Benutzbarkeitstests unterzogen und entsprechend überarbeitet.

Das Projekt **Browsing Icons** von Matthias Mayer stellt eine Navigationshilfe dar, die das Wiederfinden von Web-Seiten auch nach mehreren Tagen oder Wochen vereinfachen soll. Das Tool zeichnet hierfür in einem Fenster neben dem Browser dynamisch einen animierten Graphen der Benutzerpfade während des Surfens im Web auf. Diese Graphen werden mit Attributen der Dokumente angereichert und nach dem Beenden einer Aufgabe oder einer Sitzung als Icons bereitgestellt. Eine am HCI-Lab der University of Maryland durchgeführte Studie hat gezeigt, dass dieses Tool das

Zurückkehren zu früher besuchten Seiten in vielen Fällen vereinfacht [19]. Scone wird in diesem Projekt als Schnittstelle zum Browser verwendet.

Der **Web-Rückspiegel** ist ein Prototyp, der in einem Fenster eine Reihe von Thumbnails der Seiten darstellt, die auf das im Browser angezeigte Dokument verweisen. Hierdurch ist es möglich, im Web „rückwärts" zu navigieren und sich so einen Überblick zu verwandten Dokumenten zu verschaffen. Das Plugin fordert über den Scone-Robot oder die Google SOAP-API [14] eine Liste von „Backlinks" an. Die laut Google relevantesten dieser Seiten werden dann als Thumbnails dargestellt.

In einem weiteren Forschungsprojekt haben wir Scone dazu verwendet, um unterschiedliche **Visualisierungen und Interaktionsformen für erweiterte XLinks** [27] prototypisch umzusetzen und in Benutzbarkeitstests zu evaluieren. Die Ergebnisse der Tests werden demnächst publiziert.

7 Verwandte Arbeiten

Wenn nur Teilaspekte von Scone betrachtet werden, so gibt es eine beträchtliche Anzahl von verwandten Projekten. Proxys sind beispielsweise seit langem bekannt und bewährt, um das Caching von Web-Dokumenten durchzuführen und so die Netzlast zu reduzieren. Sie finden sich zudem in Firewalls oder auch als Filter-Werkzeuge, um Werbung aus dem Web auszublenden. Ebenso sind Robots inzwischen eine etablierte Technik, um den Index von Suchmaschinen zu erstellen oder die Konsistenz von Links zu prüfen. Scone greift auf diese Techniken zurück und erlaubt durch ihre Kombination und Erweiterung die Realisiserung von vielen unterschiedlichen Konzepten für das Web.

WBI ist ein Proxy Framework, das im IBM Almaden Research Center entwickelt wurde. WBI implementiert das Konzept der „Intermediaries": aktive Komponenten, die an beliebiger Stelle zwischen Client und Server positioniert sein können. Sie haben die Fähigkeit, Daten zu beobachten, zu verändern oder zu generieren. WBI wurde in Java implementiert und unterstützt ebenfalls ein Plugin-Konzept. Diese Plugins setzen sich aus so genannten MEGs zusammen, die mit unterschiedlichen Aufgaben in den HTTP-Datenstrom eingeklinkt werden können [18]. Scone verwendet WBI als Grundlage für die Proxy-Funktionalität, wobei es aber in entscheidenden Teilen erweitert wurde. Eine dieser Erweiterungen – ObjectStream-basierte Plugins – wurde von IBM im Rahmen eines wissenschaftlichen Austausches übernommen.

Ein mit WBI teilweise vergleichbares Java-Framework ist *Muffin*. Es wurde als Open Source Projekt an der San Diego State University entwickelt. Muffin implementiert ein Filter-System, das Endnutzern mehr Kontrolle über die zum Browser transferierten Daten geben soll. Das Projekt beinhaltet diverse anpassbare Filter, die Web-Seiten nach Bedarf analysieren, speichern und verändern können [7]. Muffin ist flexibel, erreicht aber nicht die Performanz und Funktionalität von WBI.

Arakne, ein Projekt der Universität Aarhus, ist ein offenes kollaboratives Hypermedia-System für das Web. Arakne bietet eine den Browser erweiternde Umgebung, in der eine offene Menge von Hypermedia-Werkzeugen, genannt „Views", existiert. Diese Werkzeuge sind vom Entwickler weitgehend programmierbar und statten den Benutzer mit Navigations-, Guided Tour- und Strukturierungsmechanismen für Web-Dokumente aus. Arakne kommuniziert mit dem Internet-Explorer über eine Java-

COM-Bridge und kann so direkt auf das DOM des angezeigten Dokuments zugreifen, um es zu analysieren und zu manipulieren [6].

WebSphinx von der Carnegie Mellon University unterstützt die Entwicklung von persönlichen Web-Crawlern. WebSphinx erlaubt die einfache Erstellung von Site-spezifischen, persönlichen und migrierbaren Crawlern. Das Java-Framework bietet eine Architektur, die es gestattet, nebenläufig Dokumente zu laden. WebSphinx führt den Begriff der „Classifier" ein. Diese kleinen Programme ermöglichen auf einfache Weise die Analyse und Klassifikation von Web-Dokumenten, um so den Robot zu steuern. Eine interaktive Entwicklungsumgebung, genannt „Crawler Workbench", erlaubt Benutzern, einfache perönliche Crawler zu spezifizieren und aufzurufen, ohne Programme schreiben zu müssen. Sie unterstützt zudem unterschiedliche Visualisierungstechniken für die zusammengesuchten Dokumente [20]. Einige der Konzepte von Scones Robot wurden durch dieses Projekt inspiriert.

8 Resümee

Die vielfältigen Benutzbarkeitsprobleme des Webs sind nicht zuletzt auf die oft fehlende Evaluation von Konzepten und Systemen zurückzuführen. Die wachsende Vielfalt an Anforderungen und Anwendungen im Web macht zudem neue Techniken und Sprachen notwendig, wobei hier häufig ausgereifte Konzepte für ergonomische Benutzungsschnittstellen fehlen.

Das vorgestellte Framework Scone kann als Basis dienen, um neuartige User Interfaces für Erweiterungen des Webs prototypisch zu implementieren. Diese Prototypen bieten dann eine Evaluationsgrundlage, um Konzepte und Techniken schon während der Entwicklung in Anwendungsszenarien mit Benutzern zu testen und zu optimieren.

Scone hat sich bereits in einer Reihe unterschiedlicher Projekte bewährt. Es bietet einen breiten Funktionsumfang und läuft stabil, ist aber nicht für Endprodukte konzipiert. Der Einsatz des Frameworks hat in mehreren Projekten zu einer entscheidenden Zeitersparnis geführt, so dass mehr Aufmerksamkeit den wichtigen – und in der Informatik leider immer noch viel zu oft vernachlässigten – Benutzbarkeitstests gewidmet werden konnte. Das Framework Scone sowie weitere Abbildungen und Informationen sind unter `http://www.scone.de/` verfügbar.

Danksagungen
Wir möchten allen an der Entwicklung von Scone Beteiligten unseren Dank aussprechen, insbesondere Frank Wollenweber, Torsten Haß, Björn Stephan, Hartmut Obendorf und Matthias Mayer.

Literatur

[1] D. Abrams, R. Baecker, M. Chignell: *Information Archiving with Bookmarks: Personal Web Space Construction and Organization.* In: ACM SIGCHI 1998, LA, USA, S. 41-48

[2] Keith Andrews: *Browsing, Building, and Beholding Cyberspace,* Ph.D. Thesis, Graz University of Technology, 1996, ftp://ftp.iicm.tu-graz.ac.at/pub/keith/phd/

[3] Tim Berners-Lee, James Hendler, Ora Lassila: *The Semantic Web*. Scientific American, 284(5), Mai 2001, S. 34-43

[4] M. Bieber, F. Vitali, H. Ashman, V. Balasubramaniam, H. Oinas-Kukkonen: *Forth Generation Hypermedia: Some Missing Links for the World Wide Web*. International Journal of Human Computer Studies, Vol. 47 (1), Academic Press, 1997, S. 31-65

[5] Niels O. Bouvin: *Unifying strategies for Web augmentation.* In: Proc. 10 th ACM Hypertext Conference, Darmstadt, Feb. 1999, S. 91-100

[6] Niels O. Bouvin: *Augmenting the Web through Open Hypermedia.* Ph.D. Thesis, University of Aarhus, Denmark, 2002, http://www.daimi.aau.dk/~bouvin/Arakne/thesis.pdf

[7] Mark Boyns, 1998: *Design and Implementation of a World Wide Web Filtering System*, Masters Thesis, San Diego State University, USA, 1998

[8] Andy Cockburn, Steve Jones: *Design Issues for World Wide Web Navigation Visualisation Tools.* In: Proc. RIAO'97, McGill Univ., Canada, 1997, S. 55-74

[9] A. Cockburn, B. McKenzie, M. Jasonsmith: *Evaluating a Temporal Behaviour for Web Browsers' Back and Forward Buttons*, In: Proc. 11th WWW Conference. Honolulu, 2002

[10] Jeff Conklin: *Hypertext: An Introduction and Survey.* IEEE Computer 20 (9), September 1987, S. 17-40

[11] Martin Dodge, Rob Kitchin: *Atlas of Cyberspace*, Addison Wesley, August 2001, http://www.cybergeography.org/atlas/

[12] Steve DeRose, Eve Maler, David Orchard (eds.): *XML Linking Language (XLink).* World Wide Web Consortium Recommendation, June 2001, http://w3.org/TR/xlink/

[13] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides: Design Patterns: Elements of Reusable Object-Oriented Software, Addison Wesley Longman, 1994

[14] Google Inc.: The Google Web APIs, April 2002, http://www.google.com/apis/

[15] S. Ihde, P. Maglio, J. Meyer, R. Barrett: *Intermediary-Based Transcoding Framework*. In: IBM Systems Journal, Vol. 40 (1) 2001, New York, S. 179-192

[16] William Jones, Harry Bruce, Susan Dumais: *Keeping Found Things Found on the Web.* In: 10th Conf. Information and Knowledge Management, Atlanta, 2001, S. 119-126

[17] LinkAlarm Inc.: *Web Integrity Benchmark.* Juni 2002, http://linkquality.com/

[18] Paul Maglio, Rob Barrett: *Intermediaries Personalize Information Streams.* Communications of the ACM, Vol. 43(8), August 2000, S. 96-101

[19] Matthias Mayer, Ben Bederson: *Browsing Icons: A Task-Based Approach for a Visual Web History*. HCIL Technical Report, University of Maryland, 2001

[20] Robert C. Miller, Krishna Bharat. *SPHINX: A Framework for Creating Personal, Site-Specific Web Crawlers.* In: Proc. WWW7, Brisbane, Australia, 1998, S. 119-130

[21] Jakob Nielsen: *Usability Engineering*, Academic Press, San Diego, USA, 1993

[22] James Pitkow, R. Kipp Jones : *Supporting the Web: A Distributed Hyperlink Database System*. In: Computer Networks and ISDN Systems, Vol. 28, 1996, S. 981ff

[23] Gabriel Sidler, Andrew Scott, Heiner Wolf: *Collaborative Browsing in the World Wide Web*. In: 8th Joint European Networking Conference, Edinburgh, 1997

[24] L. Tauscher, S. Greenberg: *How People Revisit Web Pages: Empirical Findings and Implications for the Design of History Systems.* Int. J. Human Computer Studies, 47(1), Academic Press, S. 97-138

[25] Weigang Wang, Roy Rada: *Experiences with Semantic Net Based Hypermedia.* In: International Journal of Human Computer Studies, Vol. 43, 1995, pp. 419-439

[26] Harald Weinreich, Winfried Lamersdorf: *Concepts for Improved Visualization of Web Link Attributes*. Computer Networks, Vol. 33, 2000, S. 403-416

[27] H. Weinreich, H. Obendorf, W. Lamersdorf: *The Look of the Link - Concepts for the User Interface of Extended Hyperlinks.* In: Proc. Hypertext'01 , Aarhus, 2001, S. 19-28