Entwicklung eines Meta-Suchinterface für WWW-Bibliothekskataloge

Universität Karlsruhe
Fakultät für Informatik

Institut für Logik, Komplexität
und Deduktionssysteme

Studienarbeit
von
Roland Sand

Betreuer:
Prof. Dr. P. H. Schmitt
Dipl.-Inform. U. Dierolf
Dr. M. Mönnich

1. Oktober 1996

Inhaltsverzeichnis

1 Aufgabenstellung 3
2 Einführung 4
2.1 Bibliotheksverbünde und -kataloge 4
2.2 Das World Wide Web 4
2.3 HTML 5
2.4 Das Common Gateway Interface 5
2.5 Die Programmiersprache Tcl 6
3 Lösungsansätze 7
3.1 Nutzung von WWW-Recherchemöglichkeiten 7
3.2 Datenbankabfrage über normierte Schnittstellen 8
4 Die Benutzeroberfläche 9
4.1 Das Anfrageformular 9
4.2 Die Ergebnispräsentation 10
5 Das Programmkonzept 13
5.1 Entgegennahme der Suchanfrage 13
5.2 Die Konfigurationsdateien 14
5.3 Formulierung der Einzelanfragen 16
5.4 Verbindungsaufbau und Absenden der Einzelanfragen 16
5.5 Lesen der Antworten 16
5.6 Die Ausgaben an den Benutzer 17
5.6.1 Header-Information 17
5.6.2 Analyse und Ausgabe der Einzelergebnisse 17
5.6.3 Ausgabe der sortierten Liste 19
5.6.4 Beenden der Ausgabe 20
5.7 Wartungsinformationen und Logeintragung 20
6 Implementierung 21
6.1 Die Auswertung des Query-Strings 21
6.2 Die Socket-Kommunikation 21
6.3 Das Sortieren der Trefferliste 23
7 Derzeitige Realisierung 24
8 Probleme und Weiterentwicklung 25
Literaturverzeichnis 26
Anhang 27

1 Aufgabenstellung

Viele Bibliotheken bieten Verzeichnisse ihrer Medienbestände an, in denen auch über Datennetze, wie z. B. dem World Wide Web, recherchiert werden kann. Darüber hinaus haben sich einzelne Bibliotheken zu Verbünden zusammengeschlossen, die u. a. den Zugriff auf ihren Gesamtbestand ermöglichen. Wenn nun bisher z. B. ein Bibliothekar wegen einer Fernleihe an einem bestimmten Buch interessiert war, mußte er in den Katalogen einzeln recherchieren, bis sich das gewünschte Suchergebnis einstellte. Neben dem hohen Zeitaufwand mußte er sich mit zahlreichen Recherchesprachen und -systemen vertraut machen, da bisher noch kein einheitlicher Standard existiert, an den sich alle Bibliotheken halten würden. Beispielsweise werden unterschiedliche Suchfelder unterstützt, die Suchbegriffe werden durch unterschiedliche Symbole trunkiert und die Trefferlisten sehen völlig unterschiedlich aus.

Im Rahmen dieser Arbeit soll nun eine Schnittstelle für das World Wide Web implementiert werden, die die Recherche in mehreren Katalogen gleichzeitig ermöglicht. Der Benutzer bräuchte also nur noch eine Anfrage zu stellen und das System nimmt für ihn die Suche in den gewünschten Katalogen ab. Es wartet auf die Einzelergebnisse und präsentiert sie ihm als einheitliches Gesamtergebnis.

2 Einführung

2.1 Bibliotheksverbünde und -kataloge

Für die Suche nach Büchern, Zeitschriften und anderen Medien sind die Kataloge der Bibliotheksverbünde besonders attraktiv, da sie einen sehr umfangreichen Bestandsnachweis anbieten. Ihre Aufgabe ist u.a. die Sammlung der Katalogdaten der wissenschaftlichen Bibliotheken eines oder mehrerer Bundesländer und Bereitstellung dieser Daten zur Recherche. Zur Zeit gibt es folgende regionale Verbundsysteme in der Bundesrepublik Deutschland [DBI_96] (das Saarland ist an keinem Verbund beteiligt):

- Bibliotheksverbund Bayern (kurz: BVB)
- Bibliotheksverbund Berlin-Brandenburg
- Gemeinsamer Bibliotheksverbund der Länder Bremen, Hamburg, Mecklenburg-Vorpommern, Niedersachsen, Sachsen-Anhalt, Schleswig-Holstein und Thüringen (GBV)
- Hessisches Bibliotheks-Informationssystem der Länder Hessen und Rheinland-Pfalz
- Nordrhein-westfälischer Bibliotheksverbund (HBZ)
- Südwestdeutscher Bibliotheksverbund der Länder Baden-Württemberg, Rheinland-Pfalz und Sachsen (SWB)

Der größte unter ihnen ist der Bibliotheksverbund Bayern mit ungefähr 8 Mil. Titeln und 15 Mil. Bestandsnachweisen.

Daneben sind jedoch die lokalen Bestandsnachweise einzelner Bibliotheken von Bedeutung, da sie häufig mehr Funktionalität anbieten können, als dies einem Verbund möglich wäre. So ist z. B. das Recherchesystem der Karlsruher Universitätsbibliothek direkt mit dem Ausleihsystem verknüpft.

2.2 Das World Wide Web

Das World Wide Web (WWW) [JoNy_95, Klut_96, Ramm_95] ist ein Informationssystem im weltweiten Internet. Es ist 1989 vom europäischen Kernforschungszentrum in Genf entwickelt worden.

Die auf dezentralen Servern bereitgestellten Daten können Texte, Bilder, Töne und vieles mehr sein. Der Benutzer benötigt einen sogenannten Browser, um die WWW-Seiten von den Servern abzurufen und anzusehen. Die Regeln mit dem sich der Client (also der Browser) und der Server verständigen ist durch HTTP (Hypertext Transfer Protocol) festgelegt (Abb. 1).

Um die gewünschte Information finden zu können, benötigt jedes Dokument eine weltweit eindeutige Adresse, die sogenannte URL (Uniform Resource Locator). Die URL besteht aus Rechnername (*Host*), evtl.. Portadresse, Pfad und Dateiname. Wenn bestimmte Teile fehlen, ist die URL *relativ*, die Angaben beziehen sich dann auf das gerade aktuelle Dokument.

Inzwischen ist das WWW zu einem Massenmedium geworden und die weitere Entwicklung ist nicht abzusehen.

2.3 HTML

Mit HTML (Hypertext Markup Language) [Tolk_95, Koch_95, Mori_95] werden die Web-Seiten erstellt. HTML ist eine Auszeichnungssprache, d. h. die eigentlichen Textteile werden markiert, damit der Client Hinweise bekommt, wie er den Text formatieren muß.

Dazu gibt es sogenannte *Tags*, die in spitzen Klammern eingeschlossen werden. Der Stelle, die markiert werden soll, wird durch einen Anfangs- und einen Ende-Tag eingeschlossen; wenn z. B. `<H1>Einleitung</H1>` im sogenannten Quelltext steht, dann soll "Einleitung" eine Hauptüberschrift sein. Es gibt auch Tags, die alleine vorkommen; `<HR>` symbolisiert beispielsweise eine horizontale Linie. Tags können durch *Attribute* mit zusätzlicher Information versehen werden, bei `<HR SIZE=3>` handelt es sich um einen Balken in einer bestimmten Dicke.

Der Begriff *Hypertext* im Namen weist auf die Möglichkeit hin, im Dokument Verweise auf beliebige andere Ressourcen im Internet zu definieren. Dazu gibt es den *Anker*, ebenfalls ein Tag: Mit `<A HREF="http://www.ubka.uni-karlsruhe.de/">UB Karlsruhe</A>` ist dem Browser bekannt, daß er zur Homepage der Unibibliothek Karlsruhe verzweigen muß, wenn der Benutzer das Wort UB Karlsruhe angeklickt hat. Der Inhalt des Attributs HREF ist dabei die entsprechende URL.

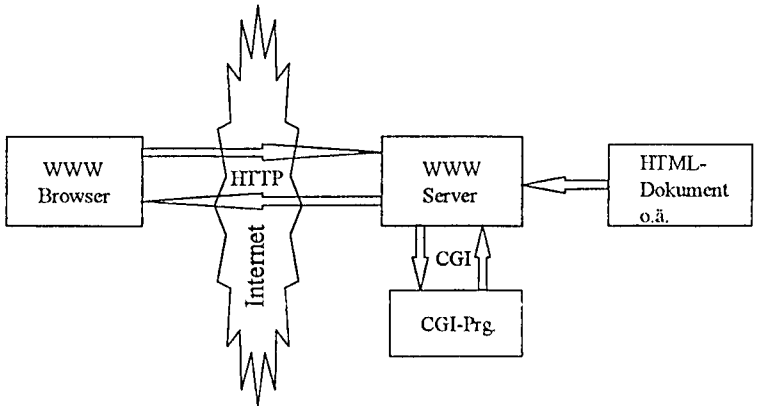

Abb. 1: Anfrage an einen WWW-Server

2.4 Das Common Gateway Interface

Das WWW kann nicht nur statische Informationen, die nur noch aufgerufen werden brauchen, bereithalten, sondern auch dynamische, also Seiten, die erst dann erstellt werden, wenn auf sie zugegriffen wird.

Möglich ist dadurch das Anbieten von Seiten, die abhängig sind von bestimmten Eingaben, z. B. einer Datenbankabfrage. HTML stellt dazu Formulare zur Verfügung, die der Benutzer an seinem Browser ausfüllen kann. Die Daten werden an den Server geschickt, der sie an ein Anwendungsprogramm weitergibt. Das Programm verarbeitet sie und erstellt als Ergebnis eine (dynamische) HTML-Seite, die es an den Server bzw. direkt an den Client zurückgibt. Die Schnittstelle zwischen Server und Anwendungsprogramm nennt man Common Gateway Interface (CGI) und das

Programm daher CGI-Programm oder CGI-Skript [Gund_96, Maur_96]. Abbildung 1 soll dies nochmals veranschaulichen.

Die Schnittstellenvereinbarung legt z. B. die Bezeichnung und den Aufbau des Inhaltes von Umgebungsvariablen fest, über die das Programm die Eingabe und weitere Informationen erhält. Die Sprache in der das Skript geschrieben ist, ist jedoch nicht vorgeschrieben. Üblich sind C, Shell-Skriptsprachen, PERL, awk und Tcl.

2.5 Die Programmiersprache Tcl

Tcl (Tool Command Language) [Oust_95, Welch_95, Maur_96] ist eine von John Ousterhout entwickelte Interpretersprache. Zwei Merkmale, die zu ihrer raschen Verbreitung sicher mit beigetragen haben, sind die leichte Erweiterbarkeit des Sprachumfangs z. B. durch Funktionen, die in einer anderen Sprache geschrieben worden sind, und ihre einfache und einheitliche Syntax, ohne den Programmierer in seinen Möglichkeiten einzuschränken. So ist beispielsweise der einzige Datentyp in Tcl die Zeichenkette. Die Interpretation als Integerzahl, als Liste oder anderes erfolgt erst zu einem späteren Zeitpunkt.

Die leichte Erweiterbarkeit von Tcl hat zu zahlreichen Ergänzungspaketen geführt, z. B. tk (toolkit), welches sehr effektiv ermöglicht, Programme mit einer grafischen Oberfläche unter dem X-Windows-System X11 zu versehen oder TclX (Extended Tcl), das die Manipulation von Dateizugriffsrechten, die Interprozeßkommunikation und vieles mehr unterstützt.

Einige Eigenschaften machen Tcl zu einer gut geeigneten CGI-Programmiersprache. Die Benutzereingabe und HTML-Ausgabe basieren ausschließlich auf dem Austausch von Zeichenketten, also dem Datentyp, auf den sich Tcl spezialisiert hat. Tcl bietet beispielsweise Befehle (`regexp` und `regsub`) an, mit denen sich Strings durch reguläre Ausdrücke erkennen und ersetzen lassen. Dies kann z. B. dazu eingesetzt werden, um Sonderzeichen bei der Eingabe, die laut CGI -Spezifikation durch einen numerischen Wert ersetzt werden mußten, zu finden und wieder auszutauschen.

Da Tcl eine Interpretersprache ist, können kleine Änderungen schnell vorgenommen und ohne vorausgehenden Übersetzungsvorgang getestet werden, und eine neue Prozedur kann auch mal im Dialogbetrieb ausprobiert werden. Der Nachteil, daß Sprachen, die erst zur Laufzeit interpretiert werden, grundsätzlich langsamer sind, ist hier nicht so entscheidend, da rechenintensive Programmabläufe in CGI-Skripten nur selten vorkommen.

Aus diesen Gründen wurde Tcl für diese Studienarbeit als Implementierungssprache ausgewählt. Außerdem wird von einigen Prozeduren aus dem TclX-Paket Gebrauch gemacht (vor allem von solchen, die die Socketkommunikation unterstützen).

3 Lösungsansätze

3.1 Nutzung von WWW-Recherchemöglichkeiten

Viele Bibliotheken und Verbünde, wie beispielsweise die Universitätsbibliothek Karlsruhe oder der Südwestdeutsche Bibliotheksverbund bieten bereits über das World Wide Web Zugang zu ihren Datenbanken. Der Benutzer muß dazu ein HTML-Formular ausfüllen, indem er einige bekannte Angaben über das oder die gesuchten Bücher angibt, z. B. Titelstichwörter, Autor, Verlag oder ISBN. Existieren Datenbankeinträge, die mit diesen Angaben übereinstimmen, wird (von einem CGI-Programm) eine "Trefferliste" erstellt und ausgegeben, die meist einige Kurzangaben über die gefundenen Bücher enthalten. Der Benutzer hat in der Regel die Möglichkeit durch "Anklicken" eines Treffers an weitere bibliographische Information zu gelangen.

Diese WWW-Schnittstelle kann nun dazu genutzt werden, um durch automatische Anfragen an mehrere Katalogen einen umfassenderen Überblick zu bekommen. Dazu muß ein Algorithmus das "Von-Hand-Ausfüllen" mehrerer Formulare unterschiedlichen Aufbaues anhand einer einmaligen Benutzereingabe simulieren, auf die Ergebnisse warten, sie aufbereiten und dem Benutzer zurückgeben (Abb. 2).

Die Aufgabe bestand nun aus der Entwicklung eines solchen Algorithmus als CGI-Programm und der Gestaltung des HTML-Formulars für die Benutzereingaben.

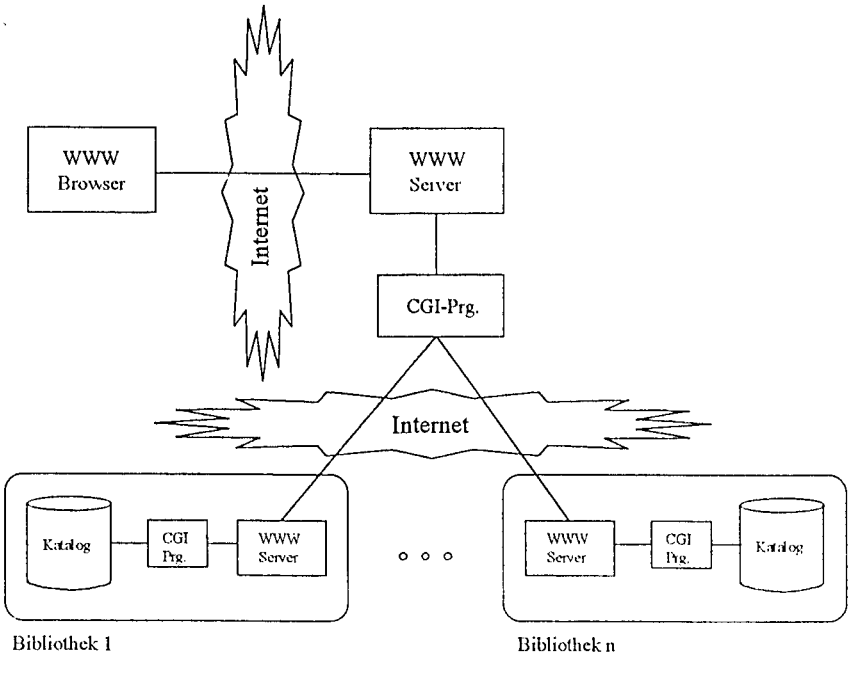

Abb. 2: CGI-Programm als Schnittstelle zwischen Benutzer und Bibliothekskatalogen

3.2 Datenbankabfrage über normierte Schnittstellen

Vereinbarungen über Schnittstellen zur Recherche in bibliographischen Datenbanken gibt es bereits. Die ISO (International Standard Organisation) hat in der SR-Definition (Search-and-Retrieve-Protokoll) einen Standard[1] beschrieben, der Datenbankzugriffe in offenen Systemen ermöglicht [GoMö_93]. SR ist ein *Application Service Element* (ASE) der Anwendungsschicht (Schicht 7) des ISO/OSI-Basisreferenzmodell, und kann somit zur Erfüllung der eigenen Aufgaben auf andere Dienste dieser Schicht oder der Schicht darunter in Anspruch nehmen. So nutzt SR beispielsweise den Austauschstandard ASN.1 (Abstract Syntax Notation One) der Darstellungsschicht, um (z. B. zwischen Rechnersystemen mit verschiedenen Zeichencodierungen) die gewünschten Funktionen zu erbringen.

Die Dienste, die SR selbst anbietet, sind u. a. Verbindungsauf- und abbau, Suchanfrage und Rückgabe des Suchergebnisses, Einfügen, Ändern und Löschen von Datensätzen, sowie Mechanismen zur Zugangskontrolle. Wegen der Normierung dieser Schnittstelle, wäre es z. B. sehr einfach, die Einzelergebnisse bei einer Suche in mehreren Datenbank zu einem Gesamtüberblick zusammenzufassen, da bereits die Teilergebnisse einheitlich sind.

Jedoch haben z. Zt. nur einige Bibliotheken diese Schnittstelle implementiert. Eine umfassende Recherche ist (noch) nicht möglich. Für eine kurzfristige Lösung ist also dieser Weg ungeeignet. Nachteilig ist außerdem, daß die zusätzlichen Funktionen, die an der WWW-Schnittstelle angeboten werden (Bestandsnachweise, Ausleihe, multimediale Zusatzinformationen, ...) und an den Endbenutzer weitergegegeben werden können, hier nicht zur Verfügung stehen.

Denkbar wäre jedoch die Integration von SR-Anfragen in das Gesamtsystem, um damit auch Bibliotheken in die Recherche mit einzubeziehen, die keinen WWW-Zugriff auf ihre Datenbank anbieten.

[1] ISO 10162-Service Definition und ISO 10163-Protocol Spezification

4 Die Benutzeroberfläche

Damit man sich ein besseres Bild des Gesamtsystems machen kann, soll zuerst auf die Benutzeroberfläche eingegangen werden. Dem Anwender wird ein Anfrageformular zur Verfügung gestellt, und das Suchergebnis sollte geeignet dargestellt werden.

4.1 Das Anfrageformular

Damit der Benutzer eine Suchanfrage im WWW stellen kann, muß er ein Formular ausfüllen und die Anfrage abschicken. Diese Seite (Abb. 3) besteht zunächst aus einigen sogenannten "Checkbuttons", mit denen aus mehreren Katalogen ausgewählt werden kann, welche bei der Suche berücksichtigt werden sollen. Zur Erzeugung u. a. solcher markierbarer Felder gibt es in HTML den Input-Tag. In diesem Fall müßte im Quelltext z. B. stehen:

```
<INPUT TYPE="checkbox" NAME="kataloge" VALUE="BLB" CHECKED>
Badische Landesbibliothek
```

Als nächstes muß sich der Anwender entscheiden, ob er am Ende eine alphabetisch sortierte Liste über alle Treffer haben möchte, da die Treffer zunächst nach Katalogen geordnet präsentiert werden. Hier ist der Aufbau des Tags ähnlich, nur handelt es sich diesmal um den Typ "radio", weil maximal ein Kästchen gleichzeitig markiert werden darf.

Nun erfolgt die eigentliche Suchanfrage. Als Suchfelder wurden die gängigsten ausgewählt; dies sind: Titel, Autor, Schlagwort, Körperschaft, ISBN (Internationale Standardbuchnummer), ISSN (International Standard Serial Number), Verlag und Erscheinungsjahr. Der Eingabetyp ist hier "text". Wenn im Quelltext für die Titeleingabe also z. B. `<INPUT TYPE="text" NAME="TI">` steht, und der Benutzer "unix" eingibt, wird dem CGI-Programm `TI=unix` übermittelt.

Schließlich folgen noch zwei Buttons um die Suchanfrage abzuschicken, und um die Felder bei Bedarf wieder zu löschen.

Damit der Browser weiß, wohin er die Eintragungen senden muß, wird ihm die URL mitgeteilt. Dazu gibt es das Form-Tag, welches das ganze Formular umschließt, z. B. `<FORM ACTION="suche.cgi" METHOD="POST"> ... </FORM>`. Die Methode "post" besagt, das die Benutzereingaben dem CGI-Skript als Standardeingabe zukommen sollen. Alternativ wäre "get" möglich, dann wird die Eingabe an die URL angehängt und das Programm kann über die Umgebungsvariable `query_string` darauf zugreifen.

Dem Formular folgen noch einige Hinweise, die der Anwender beachten sollte, z. B. mit welchem Symbol die Rechtstrunkierung erfolgt und der E-Mail-Adresse eines Ansprechpartners.

Abb. 3: Das Anfrageformular

4.2 Die Ergebnispräsentation

Das CGI-Programm liefert ein dynamisch erstelltes HTML-Dokument zurück, das die Trefferliste enthält. Im folgenden Beispiel (Abb. 4) wurde angenommen, daß der Benutzer in zwei Katalogen nach "Tcl" im Titel gesucht hat. Zunächst bekommt er die beiden Einzelergebnisse in einer annähernd einheitlichen Form präsentiert. Klickt er auf die Numerierung ("Links" sind hier durch Unterstreichung zu gekennzeichnet), verzweigt er zum betreffenden Bibliotheks-Server direkt, der ihm eine detailliertere Auskunft (Abb. 5) über den gewählten Titel liefert. Es folgen jeweils noch die Angaben, wieviel Treffer insgesamt gefunden und wieviel angezeigt wurden. Wenn mehr gefunden als angezeigt wurde, wird ein Link zu weiteren Treffern angeboten, soweit der Zielkatalog dies vorsieht.

Universitätsbibliothek Karlsruhe
KVK-Ergebnisanzeige

Südwestdeutscher Bibliotheksverbund

1 Praktisches Programmieren mit Tcl und Tk / Welch, Brent B. 1996
2 Entwurf und Implementierung einer Tcl-Sch* / Cordes, Christoph 1993
3 Tcl and the Tk toolkit / Ousterhout, John * 1995
4 Sprachenband TCL-/OXFORD-PASCAL / bearb*
5 Practical programming in Tcl and Tk / Welch, Brent B. 1995
6 Tcl und Tk / Ousterhout, John * 1995
7 Tcl and Tk toolkit / Ousterhout, John * 1994
8 Implementierung graphischer Benutzungsobe* / Abecker, Andreas 1993
9 Some limits of the partial-wave projected* / Haeringen, H. _va* 1979

Treffer gefunden: 9
Treffer angezeigt: 9

Katalog der UB Karlsruhe

1 HTML und CGI-Programmierung / Rainer Maurer , 1996
2 Practical programming in Tcl and Tk / Brent Welch , 1995
3 Exploring Expect / Don Libes , 1995
4 Tcl und Tk / John K. Ousterhout , 1995
5 Tcl and Tk toolkit / John K. Ousterhout , 1994
6 Entwurf und Implementierung einer Tcl-Schnittstelle zu InterViews / von Christoph Cordes , 1993

Treffer gefunden: 6
Treffer angezeigt: 6

Alle Treffer in sortierter Reihenfolge

(ausgenommen Einzeltreffer)

Entwurf und Implementierung einer Tcl-Sch* / Cordes, Christoph 1993 (SWB)
Entwurf und Implementierung einer Tcl-Schnittstelle zu InterViews / von Christoph Cordes 1993 (UBKA_OPAC)
Exploring Expect / Don Libes , 1995 (UBKA_OPAC)
HTML und CGI-Programmierung / Rainer Maurer , 1996 (UBKA_OPAC)
Implementierung graphischer Benutzungsobe* / Abecker, Andreas 1993 (SWB)
Practical programming in Tcl and Tk / Welch, Brent B. 1995 (SWB)
Practical programming in Tcl and Tk / Brent Welch , 1995 (UBKA_OPAC)
Praktisches Programmieren mit Tcl und Tk / Welch, Brent B. 1996 (SWB)
Some limits of the partial-wave projected* / Haeringen, H. _va* 1979 (SWB)
Sprachenband TCL-/OXFORD-PASCAL / bearb* (SWB)
Tcl and Tk toolkit / Ousterhout, John * 1994 (SWB)
Tcl und Tk / Ousterhout, John * 1995 (SWB)
Tcl und Tk / John K. Ousterhout , 1995 (UBKA_OPAC)
Tcl and Tk toolkit / John K. Ousterhout , 1994 (UBKA_OPAC)
Tcl and the Tk toolkit / Ousterhout, John * 1995 (SWB)

Treffer angezeigt: 20

Folgende Abkürzungen wurden verwendet:

SWB
Südwestdeutscher Bibliotheksverbund
UBKA_OPAC
Katalog der UB Karlsruhe

Fragen und Anmerkungen an: Uwe Dierolf (dierolf@ubka.uni-karlsruhe.de)

Abb 4: Die Ergebnisanzeige

Falls gewünscht, schließt sich noch eine alphabetisch sortierte Gesamttrefferliste an. Damit ist leicht zu erkennen, wer welches Buch in seinem Bestand hat und welche Alternativen es für eine mögliche Ausleihe gibt. Die Buch-Kurzbeschreibungen müssen zur Identifikation der Zielkataloge mit Kennzeichnungen ergänzt werden. Hier wurden Abkürzungen benutzt, die am Ende erklärt werden.

Auch hier kann die Detailanzeige angewählt werden, als anklickbare Fläche wurde die gesamte Trefferanzeige genommen. Welche Art übersichtlicher und benutzerfreundlicher ist, muß sich in der Praxis noch herausstellen.

Wenn vom Zielkatalog keine Trefferliste erkannt wird, wird einfach alles - ohne erneute Formatierung - dem Benutzer angezeigt. Dies ist z. B. bei einer Fehlermeldung der Fall, oder falls bei nur *einem* Treffer sofort die Detailanzeige angezeigt wird.

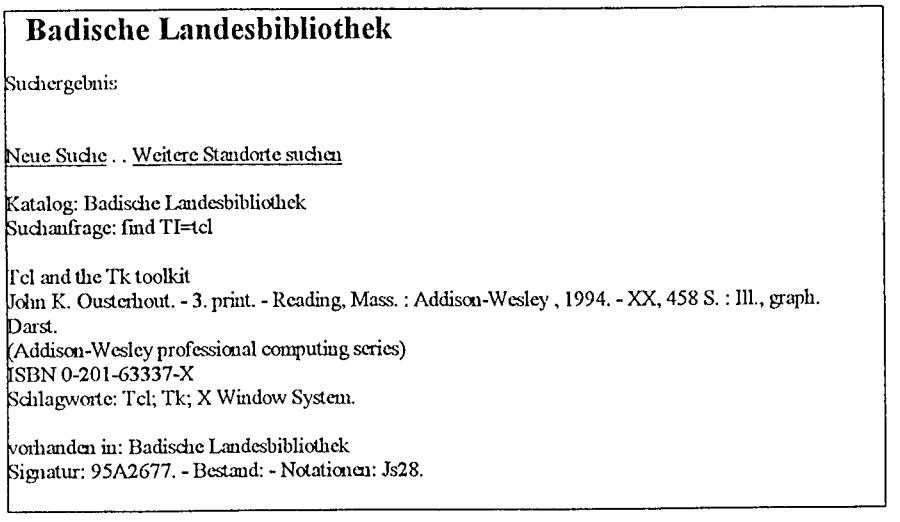

Abb. 5: Eine Ergebnis-Detailanzeige

5 Das Programmkonzept

Das Struktogramm in Abb. 6 soll den prinzipiellen Aufbau des Programms verdeutlichen. Auf die einzelnen Schritte wird in den folgenden Unterkapiteln eingegangen.

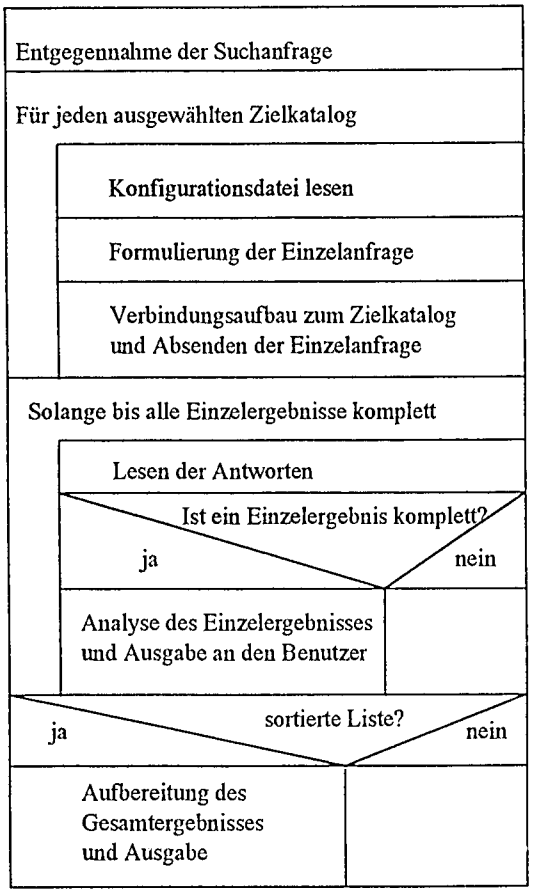

Abb. 6: Aufbau des Programms

5.1 Entgegennahme der Suchanfrage

Zunächst müssen die Daten, die der Benutzer in das Formular eingetragen hat, vom Programm gelesen werden. Die Eingaben sind von dem WWW-Browser zu einem sogenannten Query-String zusammengefaßt worden. Er hat beispielsweise folgendes Aussehen:

```
kataloge=UBKA_OPAC&kataloge=SWB&sortiert=ja&TI=program%3F+tcl&
AU=&ST=&SB=&SS=&CI=&PU=&PY=&
```

Ein Anwender möchte hier in den Katalogen der Unibücherei Karlsruhe und des Südwestdeutschen Bibliotheksverbunds nach Büchern suchen, die im Titel Begriffe beinhalten, die mit "program" beginnen (Programm, programmieren, Programmierung, program, programming, ...) und in denen zusätzlich "tcl" vorkommt.

Der Browser hat die Feldnamen, die im Formular vorkamen (kataloge, sortiert, TI, AU, ...) und die vorgegebenen Werte (UBKA_OPAC, SWB, ja), bzw. die Eintragungen vom Benutzer

("program? tcl") mit einem Gleichheitszeichen ("=") verbunden, und diese Paare wiederum mit einem kaufmännischen Und ("&") als Trennzeichen aneinander angehängt. Leergelassene Felder betrachtet er hierbei als Leerstrings.

Außerdem ersetzt der Browser einige Sonderzeichen. In diesem Fall wurde das Fragezeichen durch seine hexadezimale Darstellung ("%3F") und das trennende Leerzeichen im Titelfeld durch ein "+" ersetzt.

Das Programm muß also diese Zeichenkette lesen und daraus wieder die ursprüngliche Information gewinnen.

5.2 Die Konfigurationsdateien

Ein wesentliches Entwurfsziel war, daß das System leicht um neue Zielkataloge erweiterbar sein sollte. Zu diesem Zweck wurden die Konfigurationsdateien eingeführt.

Eine Konfigurationsdatei enthält den Programmteil, der die spezifischen Daten eines bestimmten Bibliothekskatalogs definiert. Er wird, falls benötigt, am Beginn des Programmablaufs ausgeführt. Die Bestandteile sind Variablenzuweisungen und Prozedurdefinitionen. Wenn das System um eine Datenbank erweitert werden soll, muß dafür eine solche Datei neu erstellt werden.

Die Werte, die den Variablen zugewiesen werden, sind:

- die genaue Bezeichnung der Bibliothek
- die Bestandteile der URL, mit der die Suche im Zielkatalog gestartet wird, im einzelnen:
  - der Rechnername (host)
  - der Pfad
  - die Portadresse
  - der Dateiname
- die Methode mit der die Suchanfrage gesendet wird (POST oder GET)
- das Trunkierungszeichen, das hier verwendet wird
- eine Liste mit Feldnamen, die der Katalog im Query-String erwartet
- die initialen Werte für die Feldinhalte
- eine Liste regulärer Ausdrücke, die die Struktur der Trefferzeile(n) beschreibt
- ein regulärer Ausdruck, der den Aufbau der Zeile beschreibt, die die totale Trefferzahl enthält
- ein weiterer regulärer Ausdruck, der den Aufbau der Zeile beschreibt, die einen Verweis auf weitere Treffer enthält

Die Prozeduren, die definiert werden können, sind:

- eine Prozedur, mit der das Suchergebnis eines Kataloges für die eigentliche Bearbeitung angepaßt werden kann (z. B. Einfügen von Zeilenumbrüchen)

- Prozeduren für jedes Metafeld (das sind die Felder, die in Abb. 4 auftauchen), um das entsprechende Feld, oder die entsprechenden Felder im Zielkatalog zu setzen.

Um die notwendigen Angaben für die Konfigurationsdateien herauszufinden, müssen u. a. die HTML-Quelltexte der Anfrageformulare (die ja eigentlich nur für den Browser gedacht sind) von jedem Zielkatalog untersucht werden. Folgendes Beispiel soll dies verdeutlichen:

Das Formular vom Südwestdeutschen Bibliotheksverbund (kurz: SWB) sieht in HTML ausschnittsweise so aus:

```
<FORM ACTION="/CGI/cgi-bin/swbopac.cgi" METHOD="POST">
    <SELECT NAME="kat1">
        <OPTION>Titel
        <OPTION>Schlagwort
        <OPTION>Autor
    </SELECT>
    <INPUT NAME="term1" SIZE=40 MAXLENGTH=80>
    <SELECT NAME="kat2">
        <OPTION>Titel
        <OPTION>Schlagwort
        <OPTION>Autor
    </SELECT>
    <INPUT NAME="term2" SIZE=40 MAXLENGTH=80>
</FORM>
<HR>
Rechtstrunkierung mit <B>?</B> oder <B>*</B>. Gesucht werden
die Titel, die alle Suchbegriffe enthalten
(UND-Verkn&uuml;pfung)
```

D. h. der Pfad wird in der Konfigurationsdatei auf "CGI/cgi-bin/", die Datei auf "swbopac.cgi" und die Methode auf "Post" gesetzt. Da die URL relativ ist, sind der Host und der Port dieselben, wie die aus der URL des Formulars selbst.

Beim Select-Tag handelt es sich, ähnlich wie bei den Radio-Buttons, um ein Auswahlmenü. Hier muß zunächst die Kategorie (Titel, Schlagwort oder Autor) festgelegt werden, danach erst wird das Suchfeld ausgefüllt.

Die Prozedur, die den Titel auf eine bestimmte Zeichenkette setzt, muß zuerst ein noch freies Feld suchen, dann die Kategorie (z. B. kat1) auf "Titel" setzen, und schließlich dem zugehörigen Feld für den Inhalt (term1) den gewünschten Suchbegriff zuweisen. Falls kein Feld mehr frei ist (das ist z. B. dann der Fall, wenn bereits Autor und Schlagwort gesetzt wurden), meldet die Prozedur, daß die Ausführung nicht möglich ist.

Die Feldinhalte werden von der Prozedur intern einem Array zugewiesen, der erst später, wenn alle Felder gesetzt wurden, in einen Query-String umgewandelt wird.

Entnehmen kann man dem Formular noch, wie das Trunkierungssymbol aussieht, und daß die Felder mit UND verknüpft werden. Könnte man die Art der logischen Verknüpfung in diesem Formular wählen, müßte man in der Voreinstellung die boolschen Operatoren auf UND setzen, da dies von uns vorausgesetzt wird (hätten wir im Metaformular auch ODER zugelassen, würden wir bei diesem Katalog bereits scheitern).

Die Art und Weise wie Felder gesetzt werden (ob die Kategorie gesetzt werden muß, oder wie beim Metaformular fest vorgegeben ist, wie die Feldnamen heißen und was sonst noch für Bedingungen eingehalten werden müssen) sind selbstverständlich von Katalog zu Katalog unterschiedlich. Deshalb müssen die Prozeduren für jedes Formular neu geschrieben werden und in die Konfigurationsdatei eingetragen werden.

5.3 Formulierung der Einzelanfragen

Für jede gewählte Datenbank muß eine Anfrage generiert werden. Dazu wird jeweils die Meta-Suchanfrage in die Syntax des Zielkatalogs umgesetzt.

Für jedes gesetze Feld im Metaformular ("Metafeld"), das eine Angabe über die gesuchten Daten enthält, wird für jeden Katalog das Metatrunkierungssymbol (falls vorhanden) gegen sein spezifisches Trunkierungszeichen ausgetauscht.

Mit diesem Wert wird dann die entsprechende in den Konfigurationsdateien definierte Prozedur aufgerufen, die die Umsetzung übernimmt und in einen für jeden Katalog getrennten Array abspeichert. Falls die Umsetzung eines Feldes bei einem Katalog nicht möglich ist (das Feld wird nicht unterstützt, oder kann nicht gesetzt werden, weil alle möglichen schon belegt sind) wird der Feldname in eine Fehlerliste eingetragen, die ebenfalls für jeden Katalog geführt wird.

Wenn dies für jeden Katalog erledigt ist, erfolgt die Umwandlung der Arrays in jeweils einen neuen Query-String.

5.4 Verbindungsaufbau und Absenden der Einzelanfragen

Das Programm verhält sich zu den einzelnen Bibliotheksservern als WWW-Client. Die Kommunikation erfolgt analog zu WWW-Browsern gemäß HTTP.

Zu jedem anzufragenden Katalog muß zunächst ein UNIX-Socket aufgebaut werden, dem dann die Anfrage (in welcher der Query-String enthalten ist) übergeben wird. Wichtig dabei ist, daß die HTTP-Regeln eingehalten werden und daß die richtige Anfragemethode (wie in der Konfigurationsdatei angegeben) gewählt wird.

Es ist nicht sinnvoll, die Suchanfragen einzeln zu starten und immer erst auf das Teilergebnis zu warten, da sich dann die meist ohnehin schon langen Antwortzeiten noch summieren würden. Deshalb werden erst alle Anfragen abgeschickt und danach die Antworten eingesammelt. Die Suche in den Katalogen verläuft also zeitlich parallel.

5.5 Lesen der Antworten

Nach der Anfrage wird nacheinander bei jedem Socket nachgeschaut ("Polling-Verfahren"), ob etwas zum Lesen anliegt und gegebenenfalls entgegengenommen. Da die Bearbeitungszeit vernachlässigt werden kann, beträgt die Zeit bis das Gesamtergebnis vorliegt nur solange, wie der langsamste Zielkatalog benötigt (inklusive der Laufzeit im Netz).

Sobald ein Einzelergebnis vollständig vorliegt, wird dieses sofort bearbeitet (siehe Kap. 5.6.2) und ausgegeben.

Das hat den Vorteil, das der Benutzer einen Teil des Gesamtergebnisses bereits lesen kann[2], obwohl andere Einzelergebnisse noch nicht oder erst zum Teil vorliegen.

5.6 Die Ausgaben an den Benutzer

5.6.1 Header-Information

Bevor das erste Einzelergebnis dargestellt wird, muß der HTTP- und der HTML-Header erzeugt werden. Einige Informationen für das Übertragungsprotokoll können aus den CGI-Umgebungsvariablen gewonnen werden. Außerdem wird mit dem Body begonnen und eine Gesamtüberschrift hinzugefügt. Alle Angaben, die für den Client bestimmt sind, werden einfach an die Standardausgabe geschrieben. Der Server, der den CGI-Prozeß erzeugt hat, gibt sie weiter an den Browser. Die Ausgaben im einzelnen sind z. B.:

```
HTTP/1.0 200 OK
Server: NCSA/1.4.2
Content-type: text/html

<HTML>
<HEAD>
<TITLE> KVK-Ergebnisanzeige </TITLE>
</HEAD>
<BODY>
<H1>Universit&auml;tsbibliothek Karlsruhe</H1>
<H2>KVK-Ergebnisanzeige</H2>
<HR>
```

5.6.2 Analyse und Ausgabe der Einzelergebnisse

Die Einzelergebnisse werden nacheinander in der Reihenfolge ihrer Ankunft an den Anfragenden weitergegeben. Davor ist jedoch aus verschiedenen Gründen eine Bearbeitung sinnvoll:

- Die Syntax von HTML muß eingehalten werden. Bei mehreren Einzelergebnissen hätte beispielsweise das Gesamtergebnis sonst genausoviele Header (`<HEAD>...</HEAD>`) - erlaubt ist aber nur einer.
- Relative Adressen der URLs müssen durch absolute ersetzt werden, da sie sonst vom Client mit dem Host und dem Pfad des CGI-Programms ergänzt würden (für den Browser des Anwenders ist dies das aktuelle Dokument). Relative Adressen können z. B. bei Verweisen auf Ergebnisdetailanzeigen oder bei eingebundenen Bildern auftauchen.

---

[2] Dazu muß die Ausgabe jedoch am (puffernden) Server vorbei, direkt zum Benutzer gelenkt werden. Ein CERN- oder NCSA-Server erkennt dies dadurch, das der Dateiname des CGI-Skriptes mit "nph-" (non-parsed-header) beginnt (der Name kommt daher, weil der Server am HTTP-Header keine Ergänzungen mehr machen kann).

- Damit der Benutzer leichter den Überblick behält, soll für jeden Katalog eine möglichst gleiche Trefferdarstellung verwendet werden. Ebenso soll die Information über die Gesamttrefferzahl und der Verweis auf weitere Treffer, falls vorhanden, immer in der gleichen Formulierung und an derselben Stelle erscheinen.

Deshalb wird immer dann, wenn ein Ergebnis vollständig eingetroffen ist, eine Bearbeitungs- und Ausgabeprozedur aufgerufen.

Als erstes wird als Teilüberschrift der Name des aktuellen Katalogs ausgegeben. Danach wird der HTTP-Header ausgewertet und anhand des Status festgestellt, ob die Anfrage überhaupt ausgeführt wurde. Falls dies der Fall ist, werden zur Information die Felder angezeigt, die nicht gesetzt werden konnten.

Es folgt nun im ganzen Dokument die Suche nach Treffern in der Kurzanzeige. Der Aufbau, der sich auch auf mehrere Zeilen verteilen kann, ist in der Konfigurationsdatei durch ein oder mehrere reguläre Ausdrücke beschrieben.

Als Beispiel betrachten wir einen Treffer des Bayrischen Verbundkatalogs, so wie er an uns gesendet wird (die erste Zeile wurde etwas verkürzt):

```
<LI><A HREF="/bvb-bin/isuche.cgi?uid=BIBBVB-14504100">
Schmuck aus dem modernen Japan</A>
<BR>Falk, Fritz. Pforzheim, 1983<P>
```

Die regulären Ausdrücke, die die Zeilen akzeptieren sollen, haben folgenden Aufbau:

```
^<LI><A HREF="(.*)">
^(.*)</A>
^<BR>(.*)<P>
```

"^" steht für den Zeilenanfang, "." für jedes beliebige Zeichen und "*" bedeutet, daß das vorausgehende Zeichen beliebig oft stehen darf. Mit den runden Klammern können Teilausdrücke geklammert werden, deren Inhalte weiterverarbeitet werden können. In der ersten Zeile wäre dies die URL, die zu weiterer Information führt, dann der Titel und in der letzten Zeile verschiedene Zusatzinformationen wie Autor, Ort und Erscheinungsjahr.

Wenn eine solche Treffergruppe in den Rohdaten erkannt wird, werden die Informationen in Variablen (hier: `url`, `titel` und `text`) abgelegt und diese dann in der gewünschten Formatierung wieder ausgegeben. Außerdem werden die gefundenen Treffer mitgezählt, die aktuelle Nummer sei `counter`. Der Ausgabe lautet also beispielsweise:

```
<A HREF="$url"> $counter </A> $titel. $text <BR>"
```

Der Treffer aus dem obigen Beispiel wird dann von einem Browser ungefähr so dargestellt:

1 Schmuck aus dem modernen Japan. Falk, Fritz. Pforzheim, 1983

Hinter der Nummer verbirgt sich der Verweis auf die Detailanzeige des Treffers. Da die zugehörige URL jedoch relativ ist, würde der Browser das Dokument nicht finden. Die relative URL muß also vor der Ausgabe noch durch den richtigen Rechnernamen ergänzt werden (der Pfad ist in diesem Fall jedoch mit angegeben).

Außer der Ausgabe, wird die Trefferinformation auch noch in einer Gesamtliste abgelegt, die am Ende die Treffer aller Kataloge enthält, die am Schluß sortiert ausgegeben werden soll (siehe Kap. 5.6.3).

Wenn nun Treffer gefunden und ausgegeben worden sind, kann noch mit weiteren Ausdrücken nach Zusatzinformation gesucht werden. Zum Beispiel wird nach einem Hinweis gesucht, der angibt, wieviel Treffer *insgesamt* erzielt wurden. Diese Anzahl kann höher sein als die der angezeigten Treffer, nämlich dann, wenn die Ausgabemenge des Zielkatalogs beschränkt ist. Jedoch gibt es dann meist einen Verweis auf weitere (oder alle) Treffer. Beide Informationen werden, soweit vorhanden, dem Benutzer am Ende jedes Einzeltrefferabschnitts mit ausgegeben. Zum Vergleich wird die Anzahl der eben angezeigten Treffer gegenübergestellt. Die Darstellung am Bildschirm des Benutzers sieht dann z. B. so aus:

Treffer gefunden: 42
Treffer angezeigt: 30
Weitere Treffer

Was ist aber, wenn mit Hilfe der regulären Ausdrücke keine Treffer gefunden werden? Das kann unterschiedliche Ursachen haben. Es kann sich um eine Fehlermeldung handeln, daß z. B. derzeit (aus welchen Gründen auch immer) keine Online-Recherche durchgeführt werden kann. Oder es wurden mit den Suchbegriffen tatsächlich keine Treffer gefunden. Oder es wurde genau ein Treffer erzielt, der dann sofort in einer detaillierteren Darstellung im Format des Zielkatalogs angezeigt wird. Es kann auch sein, daß sich die Darstellungsform geändert hat und die Treffer mit den bisherigen regulären Ausdrücken nicht mehr erkannt werden.

In all diesen Fällen wird der gesamte Text, der durch `<BODY>` und `</BODY>` begrenzt ist, nahezu unverändert ausgegeben. Lediglich relative Adressen müssen wiederum wie bisher durch absolute ersetzt werden. Dadurch ist auch gewährleistet, das die HTML-Regeln (falls sie nicht bereits von den Einzelkatalogen verletzt wurden!) eingehalten werden. So wird dem Benutzer keine Information vorenthalten und auf die sehr komplexe Aufgabe, Detailanzeigen und Fehlermeldungen zu analysieren wurde verzichtet. Nachteilig ist dabei die uneinheitliche Darstellung der einzelnen Kataloge in den genannten Fällen.

5.6.3 Ausgabe der sortierten Liste

Falls vom Benutzer gewünscht, wird am Ende noch eine Gesamtliste ausgegeben, die nach den Titeln alphabetisch sortiert ist. Damit soll erreicht werden, daß gleiche Bücher hintereinander erscheinen (vgl. Abb. 4). Dies kann aber trotzdem nicht immer garantiert werden, da z. B. die Titelinformation häufig unterschiedlich aufgebaut ist.

Das Zusatzangebot ist möglich, da jeder erkannte Einzeltreffer aus Kap. 5.6.2 nicht nur ausgegeben wird, sondern zusammen mit einer Kennung des betreffenden Katalogs auch in eine Gesamtliste geschrieben wird. Sie braucht dann am Ende nur noch sortiert und formatiert ausgegeben werden.

5.6.4 Beenden der Ausgabe

Schließlich werden dem Benutzer noch einige Hinweise gegeben, bevor das dynamisch erstellte HTML-Dokument durch </BODY> und </HTML> beendet wird.

5.7 Wartungsinformationen und Logeintragung

An zahlreichen Stellen im Programm können Informationen über den Ablauf in eine Wartungsdatei gespeichert werden, falls dies vom Betreiber gewünscht ist. Dies sind zum Beispiel die Eingabedaten vom Benutzer, die einzelnen Umsetzungen, die Dauer des Verbindungsaufbaus, sowie die Original-Antworten der einzelnen Kataloge. Somit kann beispielsweise eine neue Konfigurationsdatei getestet werden. Die Wartungsdatei wird bei jedem Programmablauf wieder überschrieben.

Die Datei, die die Logdaten enthält, bleibt dagegen bestehen. Bei jeder Programmausführung wird ein Datensatz angehängt. In ihr können für spätere statistische Auswertungen z. B. Datum, Uhrzeit und Ort (Rechneradresse) des Zugriffs protokolliert werden.

6 Implementierung

Anhand von einigen ausgewählten Beispielen soll nun gezeigt werden, wie Teile aus dem Konzept konkret verwirklicht wurden. Als Implementierungssprache wurde Tcl/TclX verwendet.

6.1 Die Auswertung des Query-Strings

Das Einlesen der Benutzereingaben aus einem WWW-Formular und damit auch die Auswertung eines Query-Strings (so wie in Kap. 5.1) ist eine Aufgabe, mit der sich fast jedes CGI-Programm befassen muß. Tcl-Prozeduren sind dazu in [Maur_96] und [Gund_96][3] aufgeführt.

Zunächst muß entschieden werden, nach welcher Methode die Anfrage geschickt wurde. Diese Angabe hat der Server in der Umgebungsvariablen `REQUEST_METHOD` abgelegt. Wenn es sich um die GET-Methode gehandelt hat, steht die Zeichenkette in der Umgebungsvariablen `QUERY_STRING`, ansonsten (POST-Methode) kann sie von der Standardeingabe eingelesen werden, die hierfür benötigte Zeichenkettenlänge steht in `CONTENT_LENGTH`. Am besten speichert man den Query-String in einer Programmvariablen ab, damit anschließend nicht mehr zwischen den Methoden unterschieden werden braucht.

Die nächste Aufgabe des Programmes ist die Wiederherstellung der ursprünglichen Daten. Dazu muß zuerst der Query-String bei den "&"-Zeichen aufgesplittet werden. Durch weiteres Teilen an den "="-Stellen erhält man die Variablennamen und -werte. Schließlich werden auch noch die Sonderzeichen wieder zurückersetzt.

Damit im weiteren Programmablauf darauf zugegriffen werden kann, muß die gewonnene Information geeignet abgespeichert werden. Dazu wird ein (assoziativer) Array (hier mit "fields" bezeichnet) verwendet. Die Inhalte der Feldnamen, die mehrfach auftauchen ("kataloge") werden durch Leerzeichen getrennt. Dies ist jedoch nicht bei jeder Anwendung sinnvoll. Es sollte ein Trennzeichen verwendet werden, das sonst nicht vorkommt oder man speichert die Werte in einer Liste ab.

Im Beispiel aus Kap. 5.1 müßten also folgende Zuweisungen ausgeführt werden:

```
set fields(kataloge) "UBKA_OPAC SWB"
set fields(sortiert) "ja"
set fields(TI)       "program? tcl"
set fields(AU)       ""
set fields(ST)       ""
...
```

6.2 Die Socket-Kommunikation

Die Socketunterstützung von Extended Tcl baut direkt auf der UNIX-Schnittstelle auf.

Zum Verbindungsaufbau gibt es den Befehl `server_connect`, dem die Rechneradresse und die Portnummer als Argumente übergeben werden. Rückgabewert ist ein Filehandle, der genauso

---

[3] Das angegebene Programm ist nicht ganz korrekt. Es werden nur die Sonderzeichen zurückersetzt, die in den Feldinhalten vorkommen, nicht jedoch die in den Feldbezeichnungen. Außerdem bleibt die Sonderbedeutung von "&" im `regsub`-Befehl unberücksichtigt.

für Eingabe- und Ausgabeoperation genutzt werden kann, wie der einer normalen, zum Lesen und Schreiben geöffneten Datei. Er muß einfach bei den Ein- (`gets`) und Ausgabeprozeduren (`puts`) als Argument mit angegeben werden.

Falls der Verbindungsaufbau scheitert, wird dies im Programm abgefangen, und der Filehandle wird auf einen Leerstring gesetzt. Ansonsten können nun die Einzelanfragen abgeschickt werden.

Für eine Anfrage nach der GET-Methode genügt es, wenn ein HTTP-Header an den Server gesendet wird. Wenn $path der Pfad, $file das aufzurufende Programm, $query der Anfragestring und $socket der Filehandle des entsprechenden Katalogs ist, muß folgende Anweisung ausgeführt werden:

```
puts $socket "GET $path$file?$query HTTP/1.0"
```

Der Query-String wird also einfach mit einem Fragezeichen an die Adresse angehängt. "HTTP/1.0" ist die Version des verwendeten Protokolls.

Bei der POST-Methode befindet sich der Query-String im HTTP-Inhalt, welcher vom Kopf immer durch eine Leerzeile abgetrennt ist. Der Header muß in diesem Fall durch den Typ (bei Anfragen aus einem Formular - was ja hier simuliert wird - immer so wie nachfolgend angegeben) und durch die Länge des Inhaltes ergänzt werden. Die Länge eines Strings kann in Tcl durch die Funktion `"string length"` errechnet werden. Die Anforderung sieht dann insgesamt so aus:

```
puts $socket "POST $path$file HTTP/1.0"
puts $socket "Content-Type: application/x-www-form-urlencoded"
puts $socket "Content-Length: [string length $query] \n"
puts $socket "$query"
```

Wenn alle Anfragen gestellt sind, wird auf die Antworten gewartet. Ein nützlicher TclX-Befehl um das "Polling" zu verwirklichen ist dabei `select`. Neben einigen weiteren (für unsere Zwecke uninteressanten) Aufgaben kann er aus einer Liste von Filehandles (hier also unsere Sockets), diejenige wieder zurückgeben, an denen etwas zum Lesen anliegt. Die Socket-Liste wird ihm als erstes Argument übergeben. Die nächsten zwei Übergabewerte (leere Listen) haben hier keine Bedeutung. Die Null als viertes Argument besagt, daß nicht gewartet, sondern der aktuelle Zustand sofort zurückgeben werden soll.

```
set read_list [lindex [select $socket_list {} {} 0] 0]
```

Der Rückgabewert ist eine Liste, wovon uns nur das erste Element interessiert, welches durch `[lindex Liste 0]` ausgewählt wird (Tcl beginnt Listenelemente von 0 an zu zählen). Dies ist wiederum eine Liste, bestehend aus denjenigen Sockets, an denen etwas zum Lesen anliegt, sie wird hier der Variablen `read_list` zugewiesen.

Nun wird für jedes Element von `read_list` die Eingabe gelesen und in eine für jeden Katalog getrennte Liste angehängt:

```
foreach socket $read_list {
   set zeile [gets $socket]
   lappend response($socket) $zeile
}
```

Falls eine Antwort beendet ist, wird der zugehörige Socket geschlossen und aus der `socket_list` gestrichen:

```
close $socket
set socket_list [lindex [intersect3 $socket_list $socket] 0]
```

Das Löschen eines Elements aus einer Liste wird hier durch eine Mengenoperation nachgebildet. Der `intersect3`-Befehl gibt als ersten Rückgabewert die Differenz zweier Listen zurück.

Danach wird wieder zu der Anweisung mit dem `select`-Teil gesprungen, und zwar solange, bis die Socket-Liste leer ist.

6.3 Das Sortieren der Trefferliste

Zur alphabetischen Sortierung von Zeichenketten, die Elemente einer Liste sind, gibt es den Befehl `lsort`. Die Listen-Elemente der Gesamttrefferliste, die in Kap. 5.6.3 sortiert werden sollen, sind jedoch keine (echten) Strings, sondern wiederum Listen, bestehend aus den Einzeltrefferelementen wie Titel, Autor, URL usw.

Der `lsort`-Anweisung kann man jedoch optional den Namen einer Prozedur angeben, die den Vergleich zweier Elemente durchführt und zurückgibt, welches davon bei der Sortierung zuerst kommen soll. Als Sortierkriterium sollen die Listenelemente herangezogen werden, die den Titel enthalten (hier an der ersten Stelle). Den Vergleich führt `string compare` aus, dieser liefert auch gleichzeitig den Rückgabewert, da hier der Befehl der letzte (und einzige) Bestandteil der Prozedur ist.

Zusammen mit der Prozedurdefinition müßen folgende Anweisungen ausgeführt werden:

```
proc Vergleich {a b} {
   string compare [lindex $a 0] [lindex $b 0]
}

set gesamt_liste [lsort -command Vergleich $gesamt_liste]
```

7 Derzeitige Realisierung

Das System ist unter dem Namen "Karlsruher virtueller Katalog" (KVK) am 26. Juli 1996 in das WWW-Angebot der Universitätsbibliothek Karlsruhe (http://www.ubka.uni-karlsruhe.de) aufgenommen worden. Die Kataloge in denen recherchiert werden kann, sind (Stand: September 1996):

- Katalog der Universitätsbibliothek Karlsruhe mit 320 000 Titeln
- Institutskatalog Universität Karlsruhe mit 180 000 Titeln
- Katalog der Badischen Landesbibliothek mit 400 000 Titeln
- Stuttarter Katalog (StOPAC) mit 1,1 Mio. Titeln
- Südwestdeutscher Bibliotheksverbund (SWB) mit 4,7 Mio. Titeln
- Bibliotheksverbund Bayern (BVB) mit 8 Mio. Titeln
- Verbundskatalog Nordrhein-Westfalen (HBZ) mit 6,5 Mio. Titeln
- Gemeinsamer Bibliotheksverbund (GBV) mit 6,4 Mio. Titeln
- Zeitschriftendatenbank (ZDB) des DBI
- Britischer Verbundkatalog (COPAC) mit ca. 3 Mio. Titeln
- Verzeichnis lieferbarer Bücher mit ca. 500 000 Titeln
- Koch, Neff & Oettinger & Co. GmbH Stuttgart (KNO) mit ca. 280 000 Titeln von 3000 Verlagen

Zusammen kann man also z. Zt. in einem Datenbestand von ca. 31 Mio. Titeln recherchieren. Die Anzahl der Bestandsnachweise ist noch erheblich höher.

In den ersten drei Wochen gab es bereits über 7500 Zugriffe auf den KVK. Z. Zt. (Ende September 96) erfolgen ca. 1400 Anfragen pro Tag; die Laufzeit des CGI-Skriptes beträgt dabei ca. 35 Sekunden durchschnittlich. Mit Beginn der Vorlesungszeit in den Hochschulen ist mit noch höheren Zugriffszahlen zu rechnen. Aufgrund der hohen Rechenlast mußte der Dienst bereits auf zwei WWW-Server aufgeteilt werden.

Auch die regionalen Zeitungen (siehe Anhang) und Radiostationen[4] haben sich schon für den virtuellen Katalog interessiert und darüber berichtet.

[4] "Radio Regenbogen" in der Sendung "Campus-Report" vom 27.08.96 und "Querfunk" in der Sendung vom "AK Unrad" am 22.08.96

8 Probleme und Ausblick

Keine Probleme bereiten die Zielkataloge, in denen man sich direkt auf eine Suchmaske begibt, und die Suchanfrage ohne weitere Zwischenschritte formulieren kann. Es gibt jedoch auch Systeme bei denen man sich zuerst anmelden muß, woraufhin eine Sitzungsnummer vergeben wird, die dann bei den Suchanfragen (automatisch) mit angegeben wird. Das ist z. B. bei der Zeitschriftendatenbank (ZDB) des DBI der Fall. Um dieses Problem zu lösen, hat ein Betreuer dieser Studienarbeit den KVK um Mechanismen ergänzt, die den Anmeldevorgang automatisieren, mit dem man an eine Sitzungsnummer gelangt.

Ein ähnliche Schwierigkeit tauchte beim "Gemeinsamen Bibliotheksverbund" auf. In diesem Fall wird der Port dynamisch vergeben, der in der Adresse der Suchanfrage korrekt angegeben werden muß. Die Adresse stand aber bisher unveränderlich in der Konfigurationsdatei. Auch dieses Problem konnte inzwischen vom Betreuer gelöst werden.

Insgesamt wurde durch die beiden Erweiterungen das KVK-Angebot so ausgeweitet, daß das System nun nahezu den gesamten deutschen und Teile des englischsprachigen wissenschaftlichen Buchbestand der Bibliotheken abdeckt. Dies hat zur konstant hohen Nutzung und zur allgemeinen Akzeptanz des KVK weiter beigetragen.

Eine weitere Schwierigkeit ist, daß sich der Aufbau eines Anfrageformulars oder Suchergebnisses ändern kann. Dann müßte die Konfigurationsdatei neu geschrieben werden. Nützlich wäre es, wenn weitere Mechanismen solche Änderungen automatisch erkennen, und den Betreiber darüber informieren würde. Möglich wäre dies beispielsweise durch den regelmäßigen Vergleich von bisherigen und aktuellen Suchmasken.

Wenn man z. Zt. den Querverweise folgt, z. B. zu weiteren Treffern, wird der KVK verlassen und hat auf die Formatierung keinen Einfluß mehr. Um dies zu verhindern, müßte man die enthaltenen Links durch eigene (mit Zusatzinformationen) ersetzen und den Aufruf selbst durchführen. Eine Sitzungsverwaltung wäre dabei sicherlich hilfreich.

Das System kann also sicherlich in Zukunft noch um neue Bestandteile erweitert werden, die den Datenbestand weiter vergrößern und die Benutzung und den Betrieb komfortabler machen. Es hat jedoch bereits jetzt schon einen großen praktischen Nutzen, besonders für die, die im Beruf oder in der Ausbildung regelmäßig nach Buchtiteln recherchieren müssen.

Literaturverzeichnis

[DBI_96] Deutsches Bibliotheksinstitut: Informationen zu den regionalen und überregionalen Verbundsystemen in Deutschland (5., überarb. Aufl.). Berlin, 1996

[GoMö_93] Gottswinter, Edith; Mönnich, Michael W.: "Brücken bauen zwischen EDV-Systemen: Einführung in die SR-Normen." ABI-Technik 13.1993 Nr. 4, 277-288

[Gund_96] Gundavaram, Shishir: CGI Programming on the World Wide Web. Sebastopol: O'Reilly & Associates, 1996

[JoNy_95] Jones, Russ; Nye, Adrian: HTML und das World Wide Web: Selbst publizieren im WWW. Bonn: O'Reilly/Internat. Thomson Verl., 1995

[Klut_96] Klute, Rainer: Das World Wide Web: Web-Server und -Clients, HTML 2.0/3.0, HTTP. Bonn: Addison-Wesley, 1996

[Koch_95] Koch-Steinheimer, P.: HTML: Veröffentlichen im Internet. Frankfurt: Thun, 1995

[Maur_96] Maurer, Rainer: HTML und CGI-Programmierung: Mit einer Einführung in Tcl. Heidelberg: dpunkt, 1996

[Mori_95] Morris, Mary: HTML: WWW effektiv nutzen. Hannover: Heise, 1995

[Oust_95] Ousterhout, John K.: Tcl und Tk: Entwicklung grafischer Benutzerschnittstellen für das X Windows System. Bonn: Addison-Wesley, 1995

[Ramm_95] Ramm, Frederik: Recherchieren und Publizieren im World Wide Web. Braunschweig: Vieweg, 1995

[Stev_90] Stevens, W. Richard: UNIX Network Programming. Englewood Cliffs: Prentice Hall, 1990

[Tolk_95] Tolksdorf, Robert: Die Sprache des Web: HTML 3: Informationen aufbereiten und präsentieren im Internet. Heidelberg: dpunkt, 1995

[Welch_95] Welch, Brent: Practical programming in Tcl and Tk. Upper Saddle River: Prentice Hall, 1995

Anhang

Bibliotheken „angekoppelt“

Über Internet Zugriff auf Titel von 25 Millionen Büchern

tw. Die Zahl ist fast unvorstellbar: Auf die Titel von über 25 Millionen Büchern und Zeitschriften hat man mit dem sogenannten „Karlsruher Virtuellen Katalog“ Zugriff – und das mit einer einzigen Suchanfrage im Internet. Denn die Universitätsbibliothek hat es erst- und einmalig in Deutschland geschafft, praktisch alle wichtigen deutschen Bibliotheken (deren Bestände online katalogisiert sind) sowie das Verzeichnis lieferbarer Bücher der Buchhandlungen über das Internet zu „koppeln“. „Das gibt es so in Deutschland nicht“, sagt Christoph-Hubert Schütte, Leiter der Universitätsbibliothek, nicht ohne Stolz.

Über die Homepage der Unibibliothek im Internet (http://www.ubka.uni-karlsruhe.de) haben die Benutzer jetzt mit dem „Karlsruher Virtuellen Katalog“ einen Zugang zu folgenden Bibliotheken: Die Olix-Kataloge der Unibibliothek, der Badischen Landesbibliothek und der Institutsbibliotheken der Fridericiana, Südwestdeutscher Bibliotheksverbund mit 4,7 Millionen Titeln aus Baden-Württemberg, Rheinland-Pfalz und Sachsen. Bayerischer Verbund mit acht Millionen Titeln. Verbunddatenbank des Hochschulbibliothekzentrums mit 6,5 Millionen Titeln aus Nordrhein-Westfalen und Rheinland-Pfalz. Gemeinsamer Bibliotheksverbund mit 6,4 Millionen Titeln aus Niedersachsen, Bremen, Hamburg, Schleswig-Holstein, Thüringen, Mecklenburg-Vorpommern und Sachsen-Anhalt sowie die Zeitschriftendatenbank des Deutschen Bibliotheksinstituts. Diese neue Möglichkeit ist das Ergebnis einer Studienarbeit eines Studenten der Informatikfakultät. Aufgrund der Verflechtungen zwischen Bibliothek und Informatik hat dort ein Informatiker der Bibliothek einen Lehrauftrag. Dieser Student untersuchte nun, ob eine „Koppelung“ der eigentlich höchst unterschiedlichen Computerkataloge möglich ist. „Ich war mir nicht sicher, ob das überhaupt funktioniert“, erzählt Bibliothekschef Schütte. Doch siehe da, es funktionierte – und zwar wegen der nur minimalen, aber doch immerhin einheitlichen gemeinsamen Standards eines Internetanschlusses dieser Bibliotheken.

Damit ist es laut Schütte zum ersten Mal möglich geworden, über die Grenzen der Bundesländer hinweg, Bestände zusammenzufassen, um darin recherchieren zu können. Allerdings: online bestellen kann man noch nicht. Das funktioniert weiterhin konventionell über die Möglichkeiten der Fernleihe. Mit der neuen Recherchemöglichkeit hat die Unibibliothek offensichtlich den Bedarf getroffen. Im August verzeichnete man bis jetzt genau 5 360 Nutzungen, viele davon auch aus dem Ausland.

Badische Neueste Nachrichten, 14. August 1996

Den Büchern auf der Spur

Internet-Katalog der Uni Karlsruhe findet 25 Millionen Titel

(pm). Die Zeiten, in denen Studierende oder Wissenschaftler auf der Suche nach einem seltenen Buchtitel die Computerkataloge der verschiedenen Universitätsbibliotheken nacheinander durchforsten mußten, sind vorbei: Der neue Karlsruher Virtuelle Katalog (KVK), den die Universitätsbibliothek Karlsruhe in Zusammenarbeit mit der Fakultät für Informatik entwickelt hat, ermöglicht es bundesweit erstmals, mit einer einzigen Suchanfrage in beliebigen Katalogen simultan über das Internet zu suchen und ein einheitliches Gesamtergebnis zu erhalten.

Insgesamt bietet der KVK Zugriff auf die Titel von über 25 Millionen Büchern und Zeitschriften. Schon die Einführung von Computerkatalogen in Bibliotheken stellte in den vergangenen Jahren eine große Erleichterung gegenüber früheren Zeiten dar, als die Bibliothekskunden nur zu den Zettelkatalogen der Bibliotheken vor Ort Zugang hatten.

Wer aber beispielsweise einen weniger geläufigen Buchtitel finden wollte, mußte bislang in diversen Computerkatalogen suchen. Da die Computerkataloge häufig auf unterschiedlicher Technik basieren, mußten zudem noch verschiedene Systeme bedient werden.

Die Universitätsbibliothek Karlsruhe hilft diesem Problem nun ab: Über ihre Adresse im Internet (http://www.ubka.unikarlsruhe.de) hat man Zugriff auf den KVK, der Suchanfragen an viele Kataloge ermöglicht. So verzeichnen beispielsweise die Unibibliothek Karlsruhe und die Badischen Landesbibliothek rund 900000 Titel. Im Südwestdeutschen Bibliotheksverbund schlummern 4,7 Millionen Werke aus Baden-Württemberg, Rheinland-Pfalz und Sachsen; Bayern bietet acht Millionen Titel. Weitere 13 Millionen Titel stammen aus Bibliotheken in den anderen Bundesländern.
Zudem hat man Zugriff auf das Verzeichnis, das alle im Buchhandel erhältlichen Bücher aus deutschen Verlagen auflistet. Der Benutzer kann wählen, in welchen Katalogen gesucht werden soll.

Schwarzwälder Bote, 21. August 1996