Realisierungsvorschlag für die Implementierung einer verteilten Suchmaschine im WWW

Studienarbeit

Universität Rostock, Fachbereich Informatik

vorgelegt von Titzler, Patrick
geboren am 05.03.1973 in Rostock

Betreuer: Prof. Dr. Andreas Heuer
Dr. Ing. Holger Meyer

Inhaltsverzeichnis

Kapitel 1

Einführung

Das Internet, als weltweit größtes Computernetzwerk, hat seit der Einführung des Internet-Dienstes **World Wide Web** eine rasante Entwicklung genommen. In einer Untersuchung vom Oktober 1998 hat [NSu] 3.358.969 aktive WWW-Server gefunden. Die auf diesen Servern gespeicherten Dokumente gehen in die hunderte Millionen. Um in diesem riesigen Angebot Informationen zu suchen, reicht die herkömmliche Vorgehensweise des 'Browsens' nicht aus. Der Anwender benötigt ein spezielles Werkzeug, den **Suchdienst**, das ihm die Arbeit abnimmt, die im WWW verfügbaren Dokumente zu sichten und entsprechend seinen Anforderungen relevante Dokumente zu ermitteln. Da es eine Vielzahl von unterschiedlichen Entwicklungen auf diesem Gebiet gibt, wird in diesem Kapitel eine kleine Einführung in die Suche im WWW und im Internet im Allgemeinen gegeben. Darauf aufbauend erfolgt die Motivation der Studienarbeit, die eine dieser Entwicklungen aufgreift und einen konkreten Realisierungsvorschlag für **die Implementierung einer verteilten Suchmaschine im WWW** gibt.

1.1 Das Internet

Das Internet stellt dem Nutzer eine Vielzahl von spezialisierten Diensten zur Verfügung. Zu den traditionellen Diensten gehören:

FTP Der Dienst, der nach dem verwendeten "**F**ile **T**ransfer **P**rotocol" benannt ist, ermöglicht den unkomplizierten Datenaustausch zwischen zwei Rechnern. Im speziellen geht es darum, daß Anbieter auf FTP-Servern Dateien (Text, Binary) bereitstellen, die von jedem Nutzer (anonymous FTP) bzw. autorisierten Nutzern heruntergeladen werden können.

News Das **Use**rs **net**work ist eine Sammlung von Diskussionsgruppen zu Themen aus den verschiedensten Bereichen.

WAIS Eines der ersten Systeme, das die Volltextsuche in Dokumenten, die via Internet verfügbar sind, ermöglicht, ist WAIS (**W**ide **A**rea **I**nformation

Server). Mittels WAIS wurde es möglich, Dokumente, die auf verschiedenen Rechnern vorliegen, einfach zu durchsuchen.[1] Somit kann man sagen, daß WAIS ein Vorgänger der heute im WWW verfügbaren Suchmaschinen ist.

Gopher Der Gopher ist ein Dienst zur Lokalisierung und Lieferung von Texten und Binärdateien. Die Navigation in einer Menüstruktur bot für den Nutzer im Vergleich zu FTP eine relativ komfortable Bedienung. Dadurch, daß der Nutzer kein Wissen über die Adressen von Gopher-Servern oder Verzeichnisnamen benötigte, sondern er sich quasi in einer Baumstruktur bis zum gewünschten Dokument 'durchklicken' konnte, fand der Gopher sehr schnell eine weite Verbreitung. Erst das WWW, das durch seine besseren Konzepte den Siegeszug antrat, bot dem Nutzer den Einstieg in die Welt der multimedialen Dokumente.

Durch den stetig wachsenden Umfang an Daten, die so bereit gestellt werden, ist eine manuelle Recherche nicht mehr möglich. Deshalb gibt es für jeden Dienst diverse Suchmöglichkeiten, die dem Nutzer behilflich sind, das Gesuchte mehr oder weniger schnell zu finden.

Der Vollständigkeit halber werden hier die Dienste **Telnet**, der älteste im Internet verfügbare Dienst, der zum entfernten Einloggen auf einem anderem Rechner dient, sowie **Email** erwähnt.

1.2 Suche im Internet

In diesem Abschnitt soll lediglich ein kurzer (teilweise historischer) Überblick über die diversen Suchmöglichkeiten gegeben werden.

1.2.1 Suche in FTP-Archiven

Archie[2], ein Werkzeug zur Gewinnung, Indexierung und Bereitstellung von Informationen im Internet, ist einer der Vertreter, die die Suche in FTP-Archiven ermöglichen. Die Anfrage an das Archie-System kann via *TELNET*, *Email*, *Archie-Clienten* bzw. zeitgemäß mittels *WWW-Oberflächen* erfolgen.

Die FTP-Server erzeugen in regelmäßigen Abständen ein Inhaltsverzeichnis aller bei ihnen verfügbaren Dateien, das sie den darauf spezialisierten Suchdiensten bereitstellen. Das nach eigenen Angaben umfangreichste Angebot umfaßt *Fast FTP Search*[3] mit mehr als 78 Millionen indexierten Dateien. Der Nutzer

[1] Durch Verwendung des Vektorraum-Modells wurden 'natürlichsprachige Anfragen' möglich; der zu dieser Zeit von Retrievalsystemen üblicherweise verwendete boolesche Ansatz führte zu einer wesentlich komplexeren Befehlssprache.

[2] `http://www.bunyip.com/products/archie/`

[3] `http://ftpsearch.ntnu.no/`

kann so an einer zentralen Stelle ermitteln, welche FTP-Archive die gesuchten Dateien gespeichert haben und sich die kostengünstigste Alternative (in Bezug auf Downloadzeit) auswählen. Häufig beschränkt sich die Suche in den FTP-Archiven auf den Dateinamen und den dazugehörigen Pfad; erst durch die Einführung von Dateibeschreibungen ist eine einfachere Suche á la „gib mir alle verfügbaren Antiviren-Programme aus“ möglich. In [Ftpa] und [Ftpb] sind einige FTP-Suchdienste aufgeführt.

1.2.2 Suche im Usenet

Zur Zeit gibt es mehrere zehntausend Newsgruppen zu den unterschiedlichsten Themenbereichen. Da sich einige Newsgruppen mit der gleichen Thematik beschäftigen (wenn auch unter verschiedenen Gesichtspunkten bzw. in anderen Sprachen) reicht es nicht aus, in nur einer Newsgruppe zu suchen. Suchdienste wie *Deja News*[4] helfen dem Nutzer in den Archiven Artikel zu finden, die sich mit der gewünschten Problematik auseinandersetzen. Die Suche kann wahlweise auf bestimmte Newsgruppen, Autoren, Datumsintervalle, Subjects und Inhalte beschränkt werden.

1.2.3 Suche mit WAIS

Der 1991 eingeführte *WAIS* (**W**ide **A**rea **I**nformation **S**erver) ermöglicht die Volltextsuche in lokalen als auch netzweit verteilten Dokumenten. Es existiert eine Vielzahl von WAIS-Datenbanken die auf ein Thema spezialisiert sind. Somit kann die Suche auf einige, dem Thema zugeordnete Server beschränkt werden. Durch die Verwendung des *Vektorraum-Modells* [Pfe95b] wurde eine Bewertung[5] der Ergebnisse möglich gemacht. Eine genauere Beschreibung von *WAIS* bzw. dem Ableger *freeWAIS* ist in Kapitel 3 Abschnitt 3.4 zu finden.

1.2.4 Suche auf Gopher-Servern

Die Suche in den vorhandenen Gopher-Servern mittels *Veronica* (**V**ery **E**asy **R**odent-**O**riented **N**et-wide **I**ndex to **C**omputerized **A**rchives) beschränkte sich, wie bei Archie, auf den Datei- und Verzeichnisnamen der Gopher-Sites.

1.3 Das World Wide Web

Das World Wide Web (WWW), dessen Konzepte u.a. in [Klu96] beschrieben sind, zeichnet sich durch ein rasantes Wachstum aus. Die im WWW angebotenen Informationen sind nicht nur auf (Multimedia-)Dokumente beschränkt, die

[4] `http://www.dejanews.com/`

[5] Die Bewertung wird als *Ranking* bezeichnet.

auf den WWW-Servern gespeichert sind. Der WWW-Client (Browser) ermöglicht es dem Nutzer, die traditionellen Dienste zu nutzen, was schematisch in Abbildung 1.1 dargestellt ist. Das Konzept zur einheitlichen Adressierung von beliebigen Internet-Ressourcen (**URI**:**U**niform **R**esource **I**dentifier) gestattet den einfachen Zugriff auf die traditionellen Dienste. So kann man mittels `http://www.uni-rostock.de/` auf den WWW-Server und mittels `ftp://ftp.uni-rostock.de/` auf den FTP-Server der Universität Rostock zugreifen.

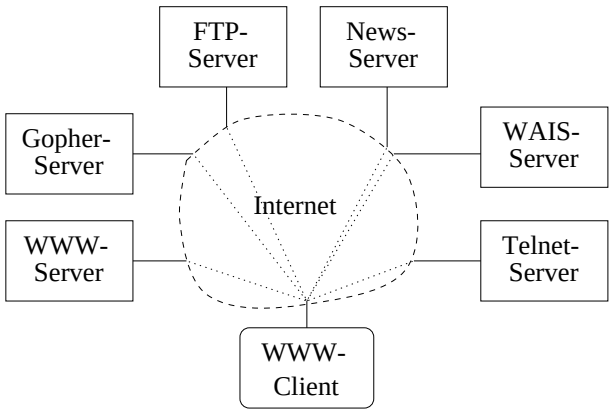

Abbildung 1.1: Integration traditioneller Internet-Dienste im WWW

1.4 Suche im WWW

Die ersten Bestrebungen, einen gewissen Überblick über interessante Dokumente im WWW zu erhalten, resultierten in langen Listen, die in mühevoller Handarbeit zusammengestellt wurden. Einer der Klassiker ist die nach dem Erfinder 'Yanoff' benannte Yanoff-Liste[6]. Die weitere Entwicklung der Suchdienste führte über die beginnende Automatisierung durch Roboter (siehe Abschnitt 1.4.2) bis zu den heute verfügbaren Suchmaschinen über Suchmaschinen (Abschnitt 1.4.3). In [Koc96] wird eine Typologie für Suchdienste eingeführt, um die Unterschiede, Möglichkeiten und Begrenzungen der verschiedenen Dienste zu verdeutlichen. Die in diesem Kapitel betrachteten Suchdienste lassen sich wie folgt einordnen:

1. Einzel-Suchdienste

   (a) Index wird manuell gesammelt bzw. angemeldet - **Kataloge** (Abschnitt 1.4.1)

   (b) Index basiert auf Daten, die von Robotern gesammelt werden - **Suchmaschinen** (Abschnitt 1.4.2)

[6] `http://sirius.we.lc.ehu.es/internet/inet.services.html`

2. Simultane Suchdienste -
   Meta-Suchmaschinen (Abschnitt 1.4.3)

Einzel-Suchdienste betrachten bei der Anfragebearbeitung nur einen Index. Im Gegensatz dazu schicken die simultanen Suchdienste die Suchanfrage an mehrere Einzeldienste gleichzeitig, bereiten die Ergebnisse auf und präsentieren sie dem Nutzer in einer einheitlichen Form. In den folgenden Abschnitten werden die einzelnen Suchdienste etwas genauer betrachtet.

1.4.1 Kataloge

Der Katalog ermöglicht dem Nutzer die Navigation in hierarchisch aufgebauten Sachgebieten. Die Einordnung von neu angemeldeten Dokumenten in diese Hierarchie erfolgt manuell; in der anschließenden redaktionellen Bearbeitung durch einen Verantwortlichen auf Seiten des Katalogbetreibers erfolgt die Verifizierung der Angaben. Durch diese Vorgehensweise wird eine höhere Qualität in der Dokumentbeschreibung, dem sogenannten Abstrakt, als bei der automatischen Erzeugung durch Suchmaschinen erreicht. Aufgrund der weit verzweigten Hierarchie der Sachgebiete bieten einige Kataloge die Möglichkeit, in den Verweisbeschreibungen und den URLs zu suchen, um so das langwierige Nachverfolgen von Verweisen (bspw. Internet → Suchen&Finden → Deutsche Suchmaschinen) zu vermeiden.

Der bekannteste Katalog, *YAHOO*[7], bietet neben dem Web-Seiten Katalog auch Informationen aus den *„Weißen Seiten“* (Adressen, Telefonnummern / nur in den USA), den *„Gelben Seiten“* (Dienstleistungsangebote), den neuesten Nachrichten (Politik/Sport/Wirtschaft/Kultur), Wetterberichten und vielen mehr. Aufgrund des vielfältigen Angebotes und den daraus resultierenden hohen Zugriffszahlen bildet YAHOO ein sehr attraktives Werbemedium für die Wirtschaft.[8] Bei der Suche in den *„Weißen Seiten“* ist die Navigation nicht vorgesehen, da das Vorgehen äquivalent zur Suche in einem Telefonbuch und dementsprechend aufwendig wäre. Im Gegensatz dazu findet bei den *„Gelben Seiten“* eine hybride Vorgehensweise Verwendung: erst die Suche nach der entsprechenden Region und anschließend die Navigation durch die Dienstleistungsangebote.

Leider besitzen Kataloge einen gravierenden Nachteil: die mangelnde Aktualität der Verweise. Um in einen Katalog aufgenommen zu werden, muß man üblicherweise seine Seiten selbst anmelden; ein automatisches Durchsuchen des WWW nach neuen Dokumenten findet, wie bereits erwähnt wurde, nicht statt. Aufrund des stark wachsenden Datenvolumens und der manuellen Bearbeitung können unmöglich alle bestehenden Katalogeinträge regelmäßig auf inhaltliche Änderungen (und daraus resultierenden Neueinordnungen) überprüft werden.

[7] `http://www.yahoo.com/` bzw. `http://www.yahoo.de/`

[8] Der derzeitige Börsenwert der Firma beträgt ca. 9 Mrd. Dollar. (Stand vom 10.Juli 1998/ Die Welt)

1.4.2 Suchmaschinen

Bei der Informationsgewinnung wählen die Suchmaschinen einen anderen Ansatz. Sie besitzen eine Komponente, den sogenannten *Robot* [Kos], die Dokumente aus dem WWW lädt[9], analysiert, Informationen (wie den Titel, Stichworte) extrahiert und in das Retrievalsystem einspeist, sowie alle vorhandenen Links rekursiv verfolgt und bearbeitet (Abbildung 1.2). In regelmäßigen Abständen besuchen

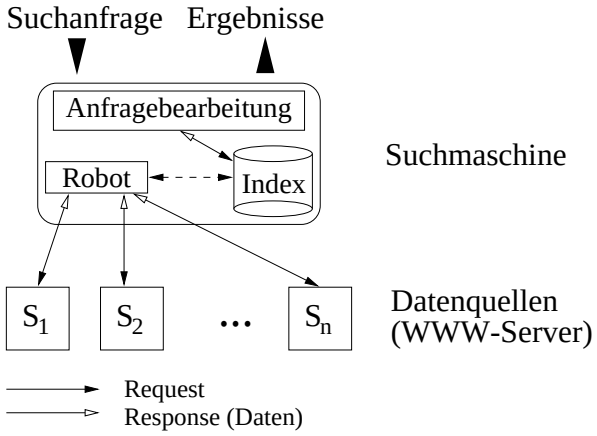

Abbildung 1.2: einfache Suchmaschine

die Robots die bereits indexierten Seiten und prüfen sie auf Veränderungen. Die automatisch gewonnenen Daten sind dabei häufig von schlechterer Qualität als die bei der manuellen Einordnung (siehe vorheriger Abschnitt) verfügbaren Informationen. Allerdings besitzt der Autor von HTML-Dokumenten die Möglichkeit seine Dokumente mit Metainformationen anzureichern, die die Suchmaschine bei der Analyse nutzen kann.[10]

Ein weiterer grundlegender Unterschied zu Katalogen besteht darin, daß die Suchergebnisse bewertet werden. Der Nutzer gibt eine natürlichsprachige Anfrage ein und erhält eine geordnete Liste, die den URL, den Dokumententitel, einen Abstrakt, die Größe, das Ranking und eventuell weitere Informationen zu diesem Dokument umfaßt. Ermöglichten ältere Suchmaschinen lediglich einfache Anfragen, so kann der Informationssuchende heute Suchbegriffe mit *booleschen Ausdrücken* (AND, OR) verknüpfen, zusammenfassen (*Phrasenbildung*) sowie zu Ergebnisdokumenten ähnliche Dokumente suchen (*relevance feedback*[11]).

[9]Welches die 'Startdokumente' sind, wird vom Betreiber des Robots festgelegt. Es besteht aber auch die Möglichkeit, daß Nutzer ihre Seiten zur Erfassung anmelden.

[10]Es gibt Suchmaschinen (z.B. *Excite* `http://www.excite.com/`), die diese Metadaten nicht nutzen.

[11]Der Suchmaschine wird vom Informationssuchenden so mitgeteilt (feedback = Rückkopplung), daß dieses Dokument aus der Ergebnismenge inhaltlich gut zur Suchanfrage paßt (relevance) und Dokumente mit ähnlichem Inhalt gesucht werden sollen, um das Ergebnis zu verbessern.

Obwohl die Suchmaschinen Millionen Web-Seiten am Tag laden, analysieren und indexieren können, reicht diese Menge nicht aus um alle im WWW vorhandenen Dokumente zu erfassen. Schätzungen über die Gesamtzahl der im WWW indexierbaren Dokumente[12] reichen bis zu 320 Millionen [LG98]. Über den Umfang der von den Suchmaschinen indexierten Dokumente (prozentual zur Gesamtzahl aller verfügbaren Dokumente) gibt es ebenfalls unterschiedliche Aussagen. Während [LG98] 34% für *HOTBOT*[13] und 28% für *AltaVista*[14] ermittelte, sind es laut [Sew] 55% bzw. 70%.

Doch nicht allein der Umfang der indexierten Dokumente bestimmt die Qualität der Suchergebnisse. Nicht selten erhält man auf eine Suchanfrage mehrere tausend Dokumentverweise zurück, von denen die wenigsten relevant sind. Durch eine genauere Spezifizierung der Suche (Verwendung von mehreren Suchbegriffen, Verknüpfungen, Phrasenbildung) läßt sich die Ergebnismenge einschränken. Häufig genug erhält man trotzdem eine nicht überschaubare Anzahl von Dokumenten. Den Grund dafür bilden die Rankingalgorithmen, die häufig auf Zählungen und Gewichtungen von Worthäufigkeiten basieren. Der Sinn des Dokumentes (die Semantik) wird an sich nicht betrachtet. Dieses Vorgehen hat zur Folge, daß eine Manipulation der Dokumente zu einer unangemessen hohen Bewertung bzw. falschen Einordnung führt.[15]

Zur Zeit kann der Nutzer mehr als 150 verfügbare Suchmaschinen befragen um seine gewünschten Informationen zu finden. Da nicht jede Suchmaschine die gleichen Dokumente indexiert hat, führt laut einer Untersuchung von [LG98] die Kombination der Ergebnisse von 6 Suchmaschinen zu einer 3.5 mal besseren Abdeckung als die einer durchschnittlichen Suchmaschine. Diesen Ansatz wählen Meta-Suchmaschinen, die im folgenden Abschnitt vorgestellt werden.

1.4.3 Meta-Suchmaschinen

Meta-Suchmaschinen sind Werkzeuge, die ein paralleles Abfragen von mehreren Suchmaschinen ermöglichen (Abbildung 1.3). In der Regel wird die Anfrage aus Performancegründen nur an wenige 'gute' Suchdienste weitergeleitet. So fragt der *MetaCrawler*[16] die Suchdienste *Lycos*, *Infoseek*, *WebCrawler*, *Excite*, *AltaVista*, *YAHOO* und *Thunderstone* ab, während sich die Meta-Suchmaschine *MetaGer*[17] auf deutsche Suchdienste spezialisiert hat. Da nicht alle angesprochenen Suchmaschinen den gleichen Funktionsumfang bieten, können die Meta-Suchmaschinen

[12] Dazu zählen keine Seiten, die als Ergebnis von Formularanfragen dynamisch erzeugt werden und Seiten, die eine Autorisierung erfordern.

[13] `http://www.hotbot.com/`

[14] `http://www.altavista.com/`

[15] Eine Möglichkeit ist das häufige Aufzählen eines Wortes. Beispiele für solche Versuche sind unter [Koc96] dokumentiert.

[16] `http://www.metacrawler.com/`

[17] `http://meta.rrzn.uni-hannover.de/`

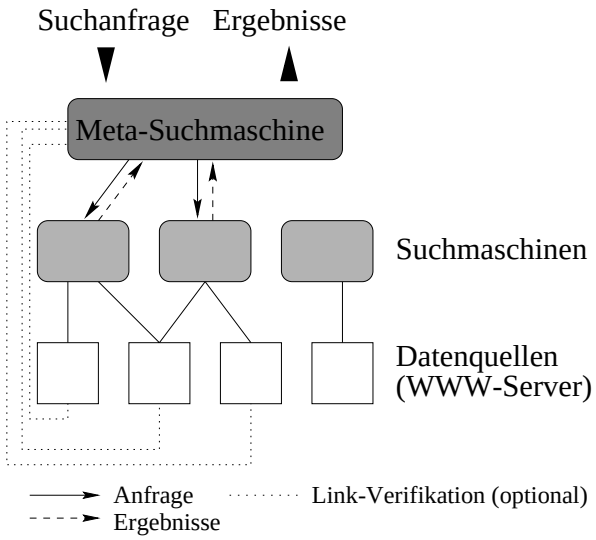

Abbildung 1.3: Meta-Suchmaschine

jeweils nur einen Teil der möglichen Funktionalität nutzen. Um ein globales Ranking, also eine einheitliche Bewertung der Ergebnisse zu ermöglichen, greifen manche Suchmaschinen auf die Dokumente selbst zurück (in Abbildung 1.3 wird dies durch die gepunktete Linie dargestellt) und umgehen so die Probleme, die bei der Zusammenfassung von Rankings, die auf unterschiedlichen bzw. unbekannten Verfahren beruhen, auftreten.

Der Funktionsumfang der Meta-Suchmaschinen variiert je nach Anbieter. Einige erlauben die (optionale) Verifikation der Ergebnis-URLs[18], andere verfügen über eine eigene Datenbasis, die sie bei Anfragen in die Suche einbeziehen.

Mit Hilfe des in [SBS98] vorgestellten Kriterienkataloges ist eine Bewertung der im WWW verfügbaren Meta-Suchmaschinen möglich geworden. Eine 'echte' Meta-Suchmaschine sollte demzufolge sechs der sieben nachfolgenden Kriterien erfüllen:

1. Parallele Suche
2. Ergebnis Merging
3. Doubletten-Eliminierung (doppelte Einträge nur einmal zusammenfassend darstellen)
4. Mindestens AND- und OR-Operatoren
5. Kein Informationsverlust (wenn ein Abstrakt existiert, muß er übernommen werden)
6. Search Engine Hiding (keine Suchmaschinen-spezifischen Kenntnisse erforderlich)

[18]Da dies ein sehr zeitaufwendiger Prozeß ist, verzichten viele auf dieses Feature.

7. Vollständige Suche (solange suchen, bis keine angesprochene Suchmaschine mehr Ergebnisse liefert)

Zur Zeit entsprechen nur der *MetaCrawler*, *Highway61*[19] und *MetaGer* diesen Anforderungen. Anhand der Kriterien ist auch erkennbar, daß die sogenannten *All-in-one-Formulare* keine Meta-Suchmaschinen sind, da die Suchdienste nur sequentiell abgefragt und die Ergebnisse demzufolge nicht gemischt werden können. Eine tabellarische Zusammenfassung aller analysierten Meta-Suchmaschinen ist in [SB98] zu finden.

1.4.4 Ausblick auf weitere Entwicklungen

Dem ständig wachsenden Datenvolumen und den daraus resultierenden riesigen Ergebnismengen müssen die heutigen Entwicklungen im Bereich der Suche im WWW Rechnung tragen.

Im Lehrgebiet *Rechnernetze und Verteilte Systeme*[20] (RVS) der Universität Hannover wird in Zusammenarbeit mit dem *Regionalen Rechenzentrum*[21] (RRZN) im Rahmen eines Teilprojektes der DFN-Expo[22] die neueste Generation (Level 3[23]) der Meta-Suchmaschinen entwickelt. Merkmal dieser neuen Generation ist, daß Level-1 und Level-2 Suchmaschinen durchsucht werden und aus den Ergebnissen eine neue Suchmaschine generiert wird.

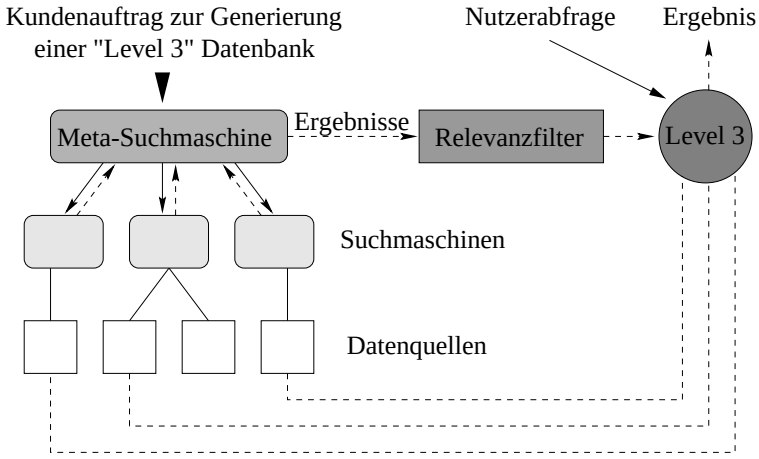

Abbildung 1.4: Meta-Suchmaschine der nächsten Generation [SB98]

Diese neuen Suchmaschinen arbeiten nach dem in Abbildung 1.4 dargestellten Prinzip. Ein Nutzer stellt an eine Meta-Suchmaschine eine Suchanfrage. Diese

[19] `http://www.highway61.com/`
[20] `http://www.rvs.uni-hannover.de/`
[21] `http://www.rrzn.uni-hannover.de/`
[22] `http://www.dfn-expo.de/`
[23] Level 1: Suchmaschinen, Level 2: Meta-Suchmaschinen

leitet die Anfrage an die zugrunde liegenden Suchmaschinen weiter und sammelt die Ergebnisse. Im anschließenden Filterprozeß werden aus den Ergebnisdokumenten diejenigen ausgewählt, die für die Recherche wirklich relevant sind und in eine Datenbank (Level 3) gespeist, die dem Nutzer für die weitere (speziellere) Recherche als Basis dient.

Im Bereich der Suchmaschinen geht die Entwicklung in Richtung der *verteilten Systeme*. In einem solchen System übernehmen Komponenten, die verteilt vorliegen, jeweils eine Teilaufgabe. Welche Komponenten dies im einzelnen sind, wird im Kapitel 2 besprochen. In dieser Studienarbeit soll eine solche verteilte Suchmaschinenarchitektur entwickelt und ein Realisierungsvorschlag für die Implementierung gegeben werden.

1.5 Motivation

Es gibt diverse Gründe, warum es sinnvoll ist, eine verteilte Suchmaschine zu entwickeln. Die Kataloge sind nicht aktuell genug und erlauben keine Bewertung der Ergebnisse. Aufgrund der rasant anwachsenden Dokumentenmenge im WWW und den daraus resultierenden Problemen ist eine zentrale Haltung der Daten über die Dokumente nicht erstrebenswert. Durch die verteilte Indexierung, Speicherung und Anfragebearbeitung zeigt sich eine Lösung dieses Problems. Im Rahmen dieser Studienarbeit erfolgt die Analyse von existierenden Suchdiensten und Informationsvermittlungsdiensten in Bezug auf ihre Einsetzbarkeit als verteilte Suchmaschine. Besonderen Wert wird auf die Aspekte der Verteilung (Indizierung, Anfragebearbeitung) und den daraus resultierenden Problemen (einheitliche Darstellung, Bewertung) gelegt. Aus den verwendeten Konzepten wird das für die jeweilige Problematik vielversprechendste ausgewählt und in die eigene Konzeption integriert. Die verwendete Anfragesprache wird so gewählt, daß Abfragen an Datenbanken, die an das WWW angeschlossen sind, möglich sind. Die weitere Studienarbeit gliedert sich wie folgt:

Kapitel 2 Es erfolgen einige *grundlegende Erläuterungen* von Begriffen und Konzepten, die in den weiteren Kapiteln verwendet werden.

Kapitel 3 Die vier betrachteten *Such- und Informationsvermittlungsdienste werden vorgestellt, analysiert und die vielversprechendsten Konzepte ermittelt.* Eine zusammenfassende Betrachtung der Erkenntnisse schafft die Voraussetzungen für den eigenen Realisierungsvorschlag, der das Kernstück der Arbeit bildet.

Kapitel 4 Ausgehend von den Ergebnissen des vorherigen Kapitels erfolgt der *Architekturentwurf* der verteilten Suchmaschine, wobei für die einzelnen Komponenten *Realisierungsvorschläge* gegeben werden.

Kapitel 5 Eine kurze *Zusammenfassung* der Ergebnisse bildet zusammen mit dem Ausblick auf offene Probleme den Abschluß der Studienarbeit.

Literaturverzeichnis Das Literaturverzeichnis referenziert Artikel, Bücher sowie online verfügbare Dokumente, auf die sich in der Studienarbeit bezogen wird.

Anhang Im Anhang werden einige Beispiele dargelegt.

Kapitel 2

Grundlagen

Zum besseren Verständnis werden in diesem Kapitel einige grundlegende Begriffe und Konzepte erläutert, die im weiteren Verlauf der Arbeit eine wichtige Rolle spielen. Ausgehend von der Beschreibung des prinzipiellen Aufbaus einer verteilten Suchmaschine in Abschnitt 2.1 werden in Abschnitt 2.2 einige mögliche Ansätze zur Verteilung der Indizes vorgestellt. Eine kurze Einführung in die den Retrievalsystemen zugrunde liegenden Modellen in Abschnitt 2.3 rundet dieses Kapitel ab.

2.1 Aufbau einer verteilten Suchmaschine

Eine verteilte Suchmaschine setzt sich aus diversen Komponenten zusammen, die jeweils eine Funktionalität bereitstellen. Zu diesen Komponenten gehören:

- **Sammelkomponente**. Diese Komponente schafft die Grundvoraussetzung für die Bereitstellung von Dokumentindizes - sie gewinnt in zyklischen Abständen aus den Dokumenten die Informationen, die von anderen Komponenten weiterverarbeitet werden. Zu den Aufgaben gehört das
  - Laden der Dokumente,
  - die Extraktion der zur Indexierung benötigten Daten (inhaltliche, wie Titel und Stichwörter, sowie Metadaten wie Größe und URL) sowie
  - die Weiterleitung der Daten an die Speicher/Indexierungskomponente.

  Für die Sammelkomponente wird der Begriff **Gatherer** eingeführt.
- **Speicher/Indexierungskomponente**. Die Speicher/Indexierungskomponente, die auch als **Retrievalsystem** bezeichnet wird, dient zur Verwaltung der indexierten Daten und zur Suche in selbigen. Die zu indexierenden Daten können der Speicher/Indexierungskomponente durch eine oder mehrere Sammelkomponenten zugeführt werden. Das Ergebnis einer Anfrage an

die Komponente ist eine Liste von bewerteten Dokumentreferenzen. Diese Bewertung wird auch als **Ranking** bezeichnet. In Abschnitt 2.3 wird ein Retrievalmodell vorgestellt, das das Ranking ermöglicht.

- **Vermittlungskomponente**. Es ist nicht unbedingt sinnvoll, ähnlich wie die Meta-Suchmaschine alle dem System bekannten Retrievalsysteme bei Anfragen zu kontaktieren. Deshalb ermittelt die Vermittlungskomponente (ggf. in Kooperation mit anderen Vermittlungskomponenten) die erfolgversprechendsten Retrievalsysteme und leitet die Suchanfrage an diese weiter. Wird die Suchanfrage von mehreren Retrievalsystemen bearbeitet, so spricht man von **verteiltem Retrieval**. Die Vermittlungskomponente wird als **Broker** bezeichnet.
- **Anfragekomponente**. Die Anfragekomponente stellt eine Schnittstelle zwischen Broker und Anwender bereit. Die Weiterleitung und Transformation der Nutzeranfrage gehört neben der Transformation und Ausgabe der Anfrageergebnisse zu den Aufgaben dieser Komponente.

Jede dieser Komponenten kann mehrfach vorkommen, um so eine bessere Performance und eine möglichst hohe Lokalität zu erreichen. Ein besonders gutes Beispiel hierfür ist der Index, der nicht zentral gehalten wird, sondern verteilt auf diversen Servern bereitgestellt werden kann. Daraus ergibt sich die Problematik, wie die Indizes zu verteilen sind. Der anschließende Abschnitt zeigt einige mögliche Ansätze.

2.2 Die Indexverteilung

Die verteilte Speicherung von Indizes bietet den Vorteil, daß die Hardwareanforderungen sinken, die Aktualität der Indizes steigt[1] und die Suche beschleunigt wird. Deshalb wird nun ein möglicher Ansatz für die Zusammenfassung von Dokument-Indizes zu Neighbourhoods (Nachbarschaften) vorgestellt. Das von [GSM97] eingeführte Konzept der **Digital Neighbourhoods** (DN) bildet eine Grundlage für die später vorzunehmende Verteilung der Indizes auf die Broker. Deshalb werden an dieser Stelle die Definitionen von DN und **Digital Distance** (DD) wiedergegeben.

Definition 1: DD by Geographic Location (DDGL).
Die DDGL zwischen zwei Dokumenten ist die physische Distanz (in km) zwischen den zwei Webservern, auf denen die Dokumente gespeichert sind.

Definition 2: DD by Network Location (DDNL).
Die DDNL zwischen zwei Dokumenten wird gemäß den IP (Internet Protocol)

[1] Da der Umfang an Indizes, die an einem Ort zu halten sind, wesentlich geringer ist als bei zentraler Speicherung und die Änderungsintervalle der Dokumente einfacher erkennbar sein dürften, kann das Update entsprechend der individuellen Anforderungen erfolgen.

Adressen der Web-Server, auf denen die Dokumente gespeichert sind, wie in Tabelle 2.1 angegeben, berechnet.

Web Server	DDNL
gleiche IP Adresse	0
gleiches Klasse C Netzwerk	1
gleiches Klasse B Netzwerk	2
gleiches Klasse A Netzwerk	3
unterschiedliche Klasse A Netzwerke	4

Tabelle 2.1: Berechnung der DDNL

Definition 3: DD by Hypertext Location (DDHTL).
Die DDHTL zwischen zwei Dokumenten ist die Länge des kürzesten bekannten Hypertextpfades zwischen diesen zwei Dokumenten im Web. Die DDHTL eines Dokumentes zu sich selbst beträgt 0.

Definition 4: DD by content (DDC).
Die DDC zwischen zwei Dokumenten wird entsprechend einer Funktion f_{DDC} : $\boldsymbol{D}oc^2 \rightarrow \boldsymbol{N}$, *wobei* $\boldsymbol{D}oc$ *die Menge aller Webdokumente ist und* $\boldsymbol{N}$ *die Menge aller natürlichen Zahlen. Die DDC zweier Dokumente mit identischem Inhalt ist Null. In Abhängigkeit von der Funktion* f_{DDC} *können auch Dokumente mit unterschiedlichem Inhalt eine Digital Distance von Null aufweisen.*

Definition 5: Digital Neighbourhood (DN) des Typs X.
Eine Digital Neighbourhood des Typs X und dem Radius R um ein Referenzdokument ist der Raum, der alle Dokumente umfaßt, deren Digital Distance des Typs X zum Referenzdokument kleiner oder gleich R ist, wobei X nur ein einzelner Typ der DD (DDGL, DDNL, DDHTL, DDC) sein kann.

Definition 6: Web View.
Ein Web View ist eine Zusammenfassung von DN, die optional eine Menge von Einschränkungen enthalten kann. Diese Zusammenfassungen werden durch die booleschen Operatoren AND und OR gebildet. Die Einschränkungen beziehen sich auf Attribute, die bei Indexierung zu jedem Dokument gespeichert werden.

Durch Einschränkungen wie „Dokumente der Sprache X“ oder „Dokumente, die in innerhalb der letzten sieben Tage geändert wurden“ läßt sich die Ergebnismenge weiter einschränken. Zum besseren Verständnis werden nun einige praktische Beispiele gegeben. Seien dn_1 und dn_2 jeweils DN, die wie folgt definiert sind:

dn_1: Typ DDC, Radius 0, Referenzdokument
`http://www.informatik.uni-rostock.de/Ordnung/ordnung.html`

dn_2: Typ DDNL, Radius 2 (gleiches Klasse B Netzwerk), Referenzdokument `http://www.uni-rostock.de/`

Dann sollte „dn_1 AND dn_2“ alle Dokumente umfassen, die auf einem Server der Uni Rostock gespeichert sind und deren Inhalt der „Vorläufigen Ordnung zur Gestaltung der WWW-Seiten des FB Informatik“ ähnelt. Dazu gehören[2] beispielsweise die Ordnung am Fachbereich Bauwesen[3] und der Wirtschafts-und Sozialwissenschaftlichen Fakultät[4].
Seien dn_3 und dn_4 jeweils DN, die wie folgt definiert sind:

dn_3: Typ DDGL, Radius 5 km,
Referenzdokument `http://www.egd.igd.fhg.de/`

dn_4: Typ DDHTL, Radius 1, Referenzdokument `http://www.uni-rostock.de/`

Durch die Verknüpfung „dn_3 AND dn_4“ wird die resultierende Ergebnismenge auf Dokumente beschränkt, die auf einem Server gespeichert sind, der sich nicht weiter als 5 km vom Server der ZGDV entfernt befindet und deren DDHTL zur Indexseite der Universität Rostock maximal eins beträgt.

In Abhängigkeit vom Anwendungsgebiet hat jedes der Konzepte seine Vorteile. Das Konzept der DDGL zur Darstellung von lokalen oder regionalen Bereichen benötigt als „Radius“ eine komplexere Form als den Kreis, um sinnvolle Anwendungsgebiete zu finden. Damit lassen sich dann auch regionale Sachverhalte wie Städte, Bundesländer oder Staaten modellieren. Eine andere Definition von Lokalität, nämlich in Bezug auf IP Adressen, liefert die DDNL. Die DDHTL nutzt die Struktur der Dokumente (Verweise, ...), die in Hypertextsystemen das Auffinden von weiteren Informationen ermöglicht. Der Inhalt von zwei Dokumenten wird durch die DDC betrachtet. Welches der Konzepte bei der Einordnung der Dokumente in die entsprechenden Digital Neighbourhood sich für die zu entwerfende Suchmaschine Verwendung findet, wird im Abschnitt 4.3.1 erläutert.

2.3 Retrievalmodelle

Schon seit langer Zeit gibt es Bestrebungen, Inhalte von Dokumenten elektronisch zu speichern und den einfachen Zugriff auf diese Inhalte zu gewährleisten. Systeme, die dieses realisieren, werden als **Information Retrieval Systeme** (Informationssysteme) bezeichnet. In den folgenden Abschnitten werden zwei Retrievalmodelle vorgestellt, auf denen diese Systeme basieren. Es handelt sich zum einen um das **Boolesche Retrieval** und das**Vektorraummodell**. Die Inhalte der nächsten Abschnitte orientieren sich am Skript der Vorlesung *Data Mining*

[2] In der Annahme, daß die Funktion f_{DDC} die DD Null liefert, wenn der Inhalt zweier Dokumente das gleiche Thema betrifft.

[3] `http://www.bau.uni-rostock.de/Document/ordnung.html`

[4] `http://www.wiwi.uni-rostock.de/HOMEPAGES/wwwordng.html`

and Information Retrieval an der Technischen Universität Darmstadt [Fer98], das eine gute Einführung in diese Thematik gibt.

2.3.1 Das Boolesche Retrieval

Die Suche in vielen Retrieval Systemen basiert auf dem Modell des 'Booleschen Retrievals'. Hierbei stellt der Nutzer an das Information Retrieval System eine Anfrage, die durch eine Verknüpfung von Paaren aus Feldbezeichnern und **Termen**[5] durch Operatoren[6] gebildet wird. Das Ergebnis einer solchen Anfrage ist eine ungeordnete Menge von Dokumenten, die die Terme in den entsprechenden Feldern enthalten. Im einfachsten Fall (`TITLE=Datenbank`) bilden diejenigen Dokumente die Ergebnismenge, die den Term `Datenbank` in der Titelzeile enthalten. Für die Verknüpfung mit der Konjunktion AND (`TITLE=Datenbank AND AUTHOR=Schulz`), der Disjunktion OR (`TITLE=Urlaub OR TITLE=Ferien`) und der Negation NOT (`TITLE=Urlaub AND NOT AUTHOR=Neckermann`) gelten die bekannten Regeln. Fehlt der Feldbezeichner in der Anfrage (`Urlaub`), nehmen viele Systeme die Vereinigung aller Felder (entspricht dem ganzen Dokument; `TITLE=Urlaub OR AUTHOR=Urlaub OR ...`) als Standardwert.

Die Implementierung von Booleschen Retrieval Systemen erfolgt gewöhnlich mit Hilfe von **invertierten Listen**. Hierbei wird eine Liste geführt, in der für jeden Term gespeichert ist in welchen Dokumenten er in welchem Feld vorkommt. Diese Vorgehensweise hat den Vorteil, daß die Suche sehr schnell abläuft. Als nachteilig erweist sich der große Speicherplatzbedarf sowie die Tatsache, daß bei Bearbeitung des Dokumentenbestandes die komplette Liste neu berechnet werden muß. Weiterhin muß entweder das Vokabular der Terme festgelegt werden, oder es müssen Regeln existieren, anhand derer ermittelt wird, ob ein Term Bestandteil des Vokabulars ist oder nicht.

Zusammenfassend betrachtet hat das Boolesche Retrieval zwei Nachteile: die Ergebnismenge ist ungeordnet und umfaßt lediglich Dokumente die die Anfragebedingungen exakt erfüllen. Als weitaus besser erweist sich das Vektorraummodell, das nun vorgestellt wird.

---

[5] Ein Term sei hier eine zusammenhängende Zeichenkette von Buchstaben, Ziffern und bestimmten Sonderzeichen. Bei näherer Betrachtung zeigt sich aber, daß diese Definition nicht unbedingt ausreichend ist. Zwar können durch Trunkierung (Autos - Auto) verschiedene Formen eines Termes zusammengefaßt werden, was aber oftmals nicht ausreicht, um alle möglichen Formen zu erkennen (Bücher - Buch). Deshalb sind andere Ansätze zu finden (z.B. der computerlinguistische Ansatz, bei dem ein Term als eine bestimmte Form eines Wortes angesehen wird) und wesentlich komplexere Verfahren (Grundformreduktion, Stammformreduktion) nötig. Der Einfachheit halber wird trotzdem die bisherige Definition beibehalten.

[6] Diese Operatoren sind AND, OR und NOT, wobei letzterer häufig nur im Zusammenwirken mit AND verwendet werden darf.

2.3.2 Das Vektorraummodell

Im Vektorraummodell werden Dokumente als Liste von gewichteten Termen dargestellt. Dadurch kann die Wichtigkeit von Termen in einem Dokument für die Beschreibung des Inhaltes berücksichtigt werden. Dies läßt sich auch auf Anfragen übertragen, d.h den Termen einer Anfrage können Gewichtungen zugeordnet werden. [Fer98] definiert das Vektorraummodell wie folgt:

Definition 7: Vektorraummodell
Sei $T = \{t_1, ..., t_n\}$ *eine endliche Menge von Termen und* $D = \{d_1, ..., d_m\}$ *eine Menge von Dokumenten. Für jedes Dokument* $d_i \epsilon D$ *sei zu jedem Term* $t_k \epsilon T$ *ein* ***Gewicht*** $w_{i,k} \epsilon R$ *gegeben. Die Gewichte des Dokumentes* d_i *lassen sich zu einem Vektor* $w_i = (w_{i,1}, ..., w_{i,n}) \epsilon R^n$ *zusammenfassen. Dieser Vektor beschreibt das Dokument im Vektorraummodell: er ist seine Repräsentation und wird* ***Dokumentvektor*** *genannt. Auch Anfragen werden durch Vektoren* $q \epsilon R^n$ *dargestellt. Wie bei der Repräsentation der Dokumente wird die Anfrage durch eine Menge gewichteter Terme, den* ***Anfragevektor*** *oder* ***Queryvektor****, dargestellt. Zwischen zwei Vektoren* $x, y \epsilon R^n$ *sei eine* ***Ähnlichkeitsfunktion*** $s : R^n \times R^n \rightarrow R$ *definiert.*

Mit Hilfe der Ähnlichkeitsfunktion lassen sich zu einem vorgegebenen Vektor ähnliche Vektoren bestimmen, um somit Dokumente mit Dokumenten bzw. Anfragen mit Dokumenten vergleichen zu können. Daraus folgt, daß bei der Suche nach Dokumenten ausgehend von einem Anfragevektor diejenigen Dokumentvektoren einer Dokumentensammlung ermittelt werden, die zum Anfragevektor ähnlich sind.
Der Ähnlichkeitswert zweier Vektoren $q = (q_1, ..., q_n) \epsilon R^n$ (repräsentiert die Anfrage) und $w_i = (w_{i,1}, ..., w_{i,n}) \epsilon R^n$ (beliebiges Dokument d_i) läßt sich mit Hilfe des Skalarproduktes berechnen:

$$s(w_i, q) = w_i \cdot q = \sum_{k=1}^{n} w_{i,k} \cdot q_k$$

Anhand der Ähnlichkeitswerte kann eine Ordnung auf den Dokumenten angegeben werden, so daß Dokumente, die viele der in der Anfrage vorkommenden Terme enthalten, am Anfang des Ergebnisses auftauchen.

Die Booleschen Operationen AND und OR lassen sich in der Annahme, daß die Gewichte der einzelnen Terme mit Null (Term kommt nicht vor) oder Eins (Term kommt mindestens einmal vor) festgelegt werden, einfach realisieren.
Anfrage in der Form $Term_l$ *AND ...* $AND\ Term_r$

$$s_{AND}(w_i, q) = \begin{cases} 1 & falls \quad w_i \cdot q = \sum_{k=1}^{n} q_k \\ 0 & sonst \end{cases}$$

Anfrage in der Form $Term_1$ *OR ...* *OR* $Term_r$

$$s_{OR}(w_i, q) = \begin{cases} 1 & falls \quad w_i \cdot q \geq 1 \\ 0 & sonst \end{cases}$$

Wie aus den Formeln ersichtlich ist, berechnet die Ähnlichkeitsfunktion anhand eines **Schwellwertes**[7] das binäre Ergebnis.

In der Regel sind die Gewichte nicht auf die Werte Eins und Null beschränkt, sondern können beliebige Werte annehmen. Die Bestimmung der Werte kann in einem manuellen oder einem (halb)automatischen Prozeß erfolgen. Die manuelle Vorgehensweise wirft aber einige Probleme auf:

- hoher Aufwand (Arbeit, Zeit, Kosten)
- geringe Konsistenz der Gewichtungen
- schlechte Einschätzung von Wahrscheinlichkeiten durch Personen

Deshalb bietet sich die (halb)automatischen Berechnung von Gewichten, die meist auf Termhäufigkeiten basiert, an. Terme, die häufig in einem Dokument vorkommen, werden stärker gewichtet; Terme, die in vielen Dokumenten vorkommen, bekommen ein geringeres Gewicht zugeordnet. Wenn Terme im Dokument für den Inhalt keine Rolle spielen, werden sie in einer **Stopwortliste**[8] erfaßt und besitzen die Gewichtung Null. Die Zusammenstellung dieser Stopwortliste ist von besonderer Bedeutung und erfolgt häufig in einem halbautomatischen Prozeß.

Ein kurzes, einfaches Beispiel soll das bisher gesagte illustrieren. Gegeben sei die Anfrage „die verteilte Suchmaschine“. Da der Artikel „die“ in der Stopwortliste auftaucht, wird er bei der Suche nicht berücksichtigt; die Anfrage wird demzufolge durch den Anfragevektor $q = (1, 1)$ repräsentiert, wobei an erster Stelle das Vorkommen des ersten Termes „verteilte“ und an letzter Stelle das Vorkommen des Termes „Suchmaschine“ aufgezählt wird. Im Retrieval System seien drei Dokumente indexiert, wobei die gleiche Gewichtung (Häufigkeit des Auftretens eines Termes im Dokument) verwendet wird. Die Repräsentation der Dokumente d_1, d_2, d_3 erfolgt durch die Dokumentvektoren $w_1 = (1, 0)$, $w_2 = (2, 3)$ und $w_3 = (0, 4)$. Die Berechnung der Ähnlichkeitswerte ergibt für das Dokument

$$d_1 : s(w_1, q) = q \cdot w_1 = 1 \cdot 1 + 0 \cdot 1 = 1$$
$$d_2 : s(w_2, q) = q \cdot w_2 = 2 \cdot 1 + 3 \cdot 1 = 5$$
$$d_3 : s(w_3, q) = q \cdot w_3 = 0 \cdot 1 + 4 \cdot 1 = 4$$

In der Ergebnisliste würde demnach das Dokument d_2 vor d_3 und d_1 stehen.

[7] Bei der AND Operation ist dieser Schwellwert maximal (Anzahl der Terme =r) und bei der OR Operation minimal (eins).

[8] im Deutschen: der, die, das, ein, ...; im Englischen: the, ...

Kapitel 3

Bewertung existierender Systeme

Nachdem ein Einblick in die Problematik des Suchens im WWW gegeben wurde und die grundlegenden Begriffe geklärt sind, erfolgt in diesem Kapitel die Betrachtung einiger konkreter Systeme. Die Wahl fiel auf die Suchdienste Harvest (Abschnitt 3.2) und freeWAIS (Abschnitt 3.4), den Informationsvermittlungsdienst von M_EDOC (Abschnitt 3.3) sowie die Spezifikation von Z39.50 (Abschnitt 3.5). Das Kapitel beginnt mit der Festlegung des verwendeten Kriterienkataloges. In den darauffolgenden Abschnitten erfolgt die Vorstellung der Systeme sowie die Analyse bezüglich der definierten Kriterien. Den Abschluß bildet die Zusammenfassung der gewonnenen Erkenntnisse, die in den Realisierungsvorschlag für den eigenen Entwurf, der im folgenden Kapitel vorgestellt wird, einfließen sollen.

3.1 Kriteriendefinition

Die Auswahl der Kriterien erfolgte in Anlehnung an [BGM96], wobei der Kriterienkatalog um die mit * markierte Kriterien erweitert und um einige für die Studienarbeit nicht relevante Kriterien gekürzt wurde. Zu diesen nicht berücksichtigten Kriterien gehören beispielsweise die Bedienbarkeit, Multimedialität und das Vorhandensein eines Profildienstes.

Datenbasis-Auswahl. Wegen des riesigen Angebotes an Datenbasen ist es nicht sinnvoll, die Suchanfrage an alle zu senden. Zum einen sind einige Angebote nicht kostenfrei bzw. haben eine lange Antwortzeit und zum anderen enthalten nicht alle Systeme relevante Dokumente. So ist es beispielsweise unsinnig in einer Datenbasis nach Dokumenten zum Thema 'Fußball' zu recherchieren, wenn diese nur 'Kochrezepte' umfaßt. Die Auswahl sollte idealerweise automatisch (und für den Nutzer transparent) erfolgen.

Heterogenität der Retrievalsysteme. Aufgrund der Vielzahl der verfügbaren Retrievalsysteme sollte der Broker eine Schnittstelle anbieten, die eine einfache Anbindung an das Retrievalsystem ermöglicht.

Integration existierender Suchdienste.* Da es bereits eine Vielzahl an Suchdiensten im WWW gibt, bietet es sich an, diese zu integrieren.

Integration von Datenbanken.* Ein Teil der im WWW bereitgestellten Dokumente liegt dynamisch vor, d.h. sie werden erst erzeugt, wenn sie von einem Nutzer angefordert werden. Dies bietet sich vor allem dann an, wenn Daten aufbereitet werden, die sich häufig ändern bzw. in denen gesucht werden kann (Online-Kataloge, Fahrpläne). Da eine Suchmaschine nicht den gesamten Inhalt einer Datenbank indexieren kann, muß ein Kompromiß gefunden werden, um trotzdem Informationen über die Datenbank für die Suchmaschine bereitzustellen. Wie dies erfolgen kann, ist unter [Web98] und [Gar98] nachzulesen.

Metadatenhaltung. Von Interesse sind u.a. die Art der Metadatengewinnung und die Aktualisierung bei Änderungen. *Dieses Kriterium wird nicht bewertet, sondern der Vollständigkeit halber erfaßt.*

Ranking. Wenn die Datenbasen ein Ranking der Ergebnisdokumente unterstützen, sind im Fall des verteilten Retrievals die Ergebnisse zu mischen und einheitlich zu bewerten. Die Problematik besteht u.a. darin, daß nicht alle Datenbasen ihre Rankingalgorithmen offenlegen bzw. auch bei gleichen Algorithmen unterschiedliche Bewertungen erzielen [GCGMP97]. Neben dem verwendeten Rankingalgorithmus müssen weitere Informationen (Anzahl der Dokumente in der Datenbasis, etc.) über die Dokumente und die Datenbasis in das globale Ranking einfließen. *Dieses Kriterium wird zweifach bewertet: lokales Ranking / globales Ranking.*

Verteiltes Retrieval. Der Nutzer muß in der Lage sein in mehreren Datenbasen zu recherchieren. Ist dies gegeben, ist die parallele Bearbeitung anzustreben, da eine sequentielle Bearbeitung die Gesamtantwortzeit unnötig verlängert.

Verteiltheit/Replikation. Besonderes Augenmerk wird auf die Verteilung der Systeme gelegt. Dabei stellt sich die Frage, welche Daten verteilt gehalten werden[1] und welche sogar repliziert vorliegen. Letzteres spielt eine besondere Rolle, wenn eine hohe Verfügbarkeit und eine gute Lastbalancierung gefordert ist.

Die in den Systemen verwendeten Konzepte werden hinsichtlich ihrer Einsetzbarkeit im eigenen Realisierungsvorschlag geprüft und wie folgt eingeordnet:

⊘ das Kriterium ist keine Anforderung an das System; es existiert demzufolge kein Konzept, das im eigenen Realisierungsvorschlag verwendet werden kann

[1] z.B. die Indizes

- $\ominus$ Kriterium nicht erfüllt; es gibt kein Konzept, das betrachtet werden kann
- $\odot$ Kriterium erfüllt; das verwendete Konzept ist nicht für den eigenen Realisierungsvorschlag geeignet
- $\oplus$ Kriterium erfüllt; das Konzept ist gut in den eigenen Vorschlag zu integrieren

Darüber hinaus werden die einzelnen Systeme auf ihre

Erweiterbarkeit/Verfügbarkeit* geprüft, d.h. es erfolgt eine Betrachtung, wie offen das System bezüglich Erweiterungen ist und wie gut die Verfügbarkeit von Informationen und Hilfen (Newsgruppen, FAQ, Dokumentation) zum System sind. Ausschlaggebend für die Verfügbarkeit ist, daß Programmquellen oder eigenständige Komponenten frei zugänglich sind, um sie im eigenen Realisierungsvorschlag nutzen zu können. Eine weite Verbreitung des Systems wirkt sich ebenfalls positiv auf die Bewertung aus, da dies eine breite Basis für Weiterentwicklungen schafft. Für die Gesamtbeurteilung bezüglich der **praktischen Verwendbarkeit als Basis im eigenen Realisierungsvorschlag** werden die beiden Kriterien zusammengefaßt. Ist mindestens eines der Kriterien nicht ausreichend erfüllt, lautet die Bewertung $\otimes$ (nicht verwendbar). Im anderen Fall (beide Kriterien erfüllt) wird das System mit $\oplus$ (verwendbar) bewertet. Die optische Unterscheidung zu den bisherigen Bewertungssymbolen wurde notwendig, um die Bedeutung des Kriteriums hervorzuheben.

3.2 Harvest

> *„We call the system Harvest to connote its focus on reaping the growing crop of Internet information.“* [BD+95]

An der Universität von Colorado wurde in Zusammenarbeit mit Wissenschaftlern anderer Hochschulen die Suchmaschine Harvest[2] entwickelt und nach Projektende zum einen als freie Software und zum anderen als Grundbaustein für kommerzielle Zwecke weiterentwickelt. Die am Fachbereich Informatik entwickelte Suchmaschine *Swing*[3] basiert beispielsweise auf dem System. Die Motivation [BDMS94] zur Entwicklung von Harvest bildeten drei Aspekte:

1. die Vielfältigkeit der Informationssysteme
2. das stark wachsende Nutzeraufkommen
3. die rapide anwachsenden Datenmengen.

Diese Aspekte spiegeln sich in der Architektur wieder, die im kommenden Abschnitt vorgestellt wird.

[2] `http://harvest.transarc.com/`
[3] `http://swing.informatik.uni-rostock.de/`

3.2.1 Architektur

Das modular aufgebaute System setzt sich aus den folgenden Komponenten (Abbildung 3.1) zusammen:

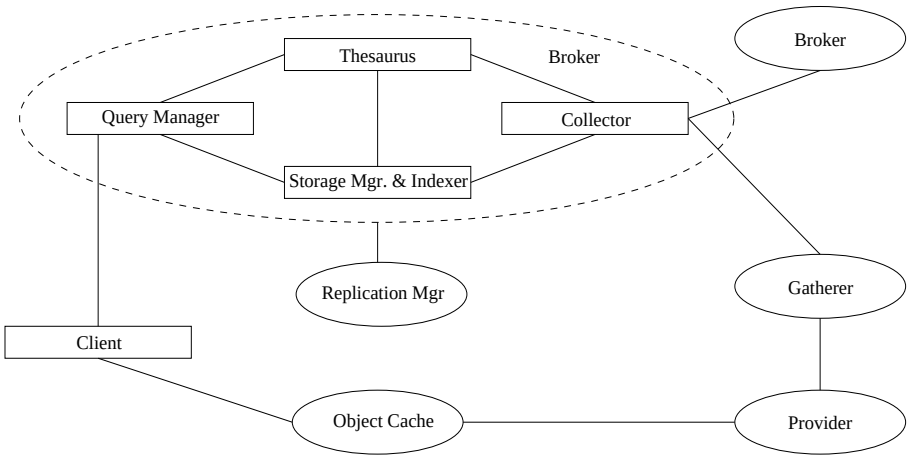

Abbildung 3.1: Harvest Architektur [HSW96]

- **Gatherer** sammeln Dokumente mittels verschiedener Zugriffsmethoden, extrahieren Metainformationen und verteilen diese an einen oder mehrere Broker. Gatherer laufen üblicherweise auf Providerseite (lokaler Zugriff auf die Dokumente) und reduzieren so die Netzbelastung im Vergleich zur herkömmlichen Vorgehensweise (Laden der Dokumente über das WWW mittels FTP, HTTP, Gopher, NNTP) auf ein Minimum. Die Übertragung der Metainformationen erfolgt in einem einzelnem Datenstrom unter Verwendung des *Summary Object Interchange Formates* (SOIF).

- **Broker** stellen das Anfrageinterface für die gesammelten Informationen zur Verfügung und rufen das Index/Search Subsystem bei Anfragen auf. Mit Hilfe eines *Bulk Transfer Protocol* können Broker auch Metainformationen, die auf anderen Brokern indiziert sind, einlesen.

- **Index/Search Subsystem.** Harvest ermöglicht durch ein allgemeines Broker-Indexer Interface die Verwendung verschiedener Retrievalsysteme. Zur Zeit werden unter anderem freeWAIS, Glimpse und Nebula unterstützt.

- **Object Cache.** Der hierarchisch aufgebaute Object Cache ermöglicht das Zwischenspeichern von HTTP-, FTP-, Gopher- und DNS-Anfrageergebnissen. Seit 1996 ist er nicht mehr Bestandteil von Harvest, sondern wird als eigenständiges Produkt unter dem Namen *Squid*[4] weiterentwickelt.

[4] `http://squid.nlanr.net/Squid/`

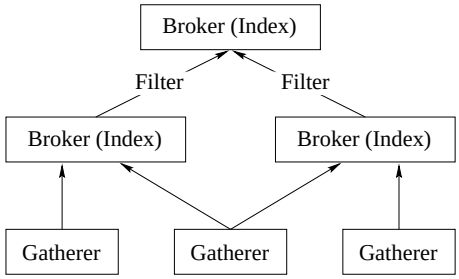

Abbildung 3.2: kaskadierte Anordnung von Brokern

- **Replicator.** Um eine möglichst gute Lastverteilung unter den Brokern zu erreichen, werden die Datenbanken der Broker repliziert. Dadurch wird die Anfragebearbeitung beschleunigt und die Verfügbarkeit verbessert.

3.2.2 Anfragebearbeitung

1. Der Nutzer stellt an den Broker eine Anfrage.
2. Der Broker ruft das Index/Search Subsystem auf, das die in Frage kommenden Dokumente ermittelt.
3. Die gefundenen Dokumentverweise werden bewertet und dem Nutzer präsentiert.

3.2.3 Kriterien

Datenbasis-Auswahl. Es besteht die Möglichkeit in verschiedenen (thematisch spezialisierten) Brokern zu suchen, wobei aber jeder Broker einzeln angesprochen werden muß.

Bewertung: $\oslash$

Heterogenität der Retrievalsysteme. Durch das flexible Broker-Indexer Interface kann bei Harvest jedes Textretrievalsystem verwendet werden, das boolesche Kombinationen von attributbasierten Anfragen und inkrementelle Updates unterstützt [Tec96].

Bewertung: $\oplus$

Integration existierender Suchdienste. Eine Integration ist nicht vorgesehen.

Bewertung: $\oslash$

Integration von Datenbanken. Harvest bietet keine explizite Unterstützung für die Integration von Datenbanken. Durch die offenen Schnittstellen des Gatherers läßt sich dieses Kriterium trotzdem erfüllen. Um Datenbanken,

bzw. dynamische Dokumente, die aus diesen generiert werden, verwenden zu können, müssen die Schemainformationen und Attributwerte in das SOIF transformiert werden.

Bewertung: ⊘

Metadatenhaltung. Die Gewinnung und Aktualisierung der Metadaten erfolgt automatisch durch den Gatherer. Durch eine kaskadierte Anordnung der Broker können verschiedene Sichten auf die Metadaten verfügbar sein (Abbildung 3.2).

Ranking. Das Retrievalsystem bestimmt das verwendete Ranking. Da kein verteiltes Retrieval stattfindet, gibt es kein Konzept zum Mischen und zur einheitlichen Bewertung der Ergebnisse.

Bewertung: ⊕ / ⊘

Verteiltes Retrieval. Die Anfragen werden bei Harvest durch den Broker bearbeitet, der nur in seiner eigenen Datenbasis suchen kann. Damit ist eine Kooperation der Broker und das verteilte Retrieval nicht möglich.

Bewertung: ⊘

Verteiltheit/Replikation. Die Komponenten (Broker, Gatherer, Cache) können sich auf unterschiedlichen Servern befinden. Die Datenbasen der Broker liegen nicht verteilt vor; jedem Broker ist 'seine' Datenbasis, das Retrievalsystem, zugeordnet. Die Gatherer halten die Informationen zu den von ihnen durchsuchten Datenbeständen lokal, um so ein inkrementelles Update zu ermöglichen. Eine Replikation der Broker ist möglich, wobei zwischen den Replikaten schwache Konsistenz gewährleistet ist.

Bewertung: ⊕

Erweiterbarkeit/Verfügbarkeit. Das Harvest-System bietet sehr gute Erweiterungsmöglichkeiten, die eine relativ einfache Integration von neuen Funktionen gestatten. Beispielhaft seien die zur Verfügung stehenden Werkzeuge zur Indexgewinnung genannt. Mit dem SOIF steht ein einfaches Austauschformat für Dokumentinformationen bereit. Insbesondere das Vorhandensein einer aktiven Newsgruppe[5] und die offen zugänglichen Quellen (Code und Dokumentation[6]) sprechen für Harvest als Basis für einen verteilten Suchdienst.

Bewertung: ⊕

[5] `news:comp.infosystems.harvest`

[6] Die Dokumentation beschränkt sich überwiegend auf Installation und das Setup der Komponenten.

3.2.4 Fazit

Die Konzepte von Harvest bieten gute Möglichkeiten, das System so zu erweitern, daß Komponenten für die eigene Konzeption genutzt werden können. Beispielsweise wäre folgendes Szenario denkbar: Durch die Einführung einer neuen Komponente, die Verteilungsaufgaben übernimmt und sich um das Zusammenfassen der Ergebnisse kümmert, könnte die verteilte Bearbeitung realisiert werden. Die Datenbasisauswahl könnte sowohl manuell als auch automatisch in dieser Komponente erfolgen. Eine Erweiterung des Broker-Indexer Interfaces schafft die Voraussetzung für die Integration von existierenden Suchdiensten, die dann praktisch die Datenbasis für einen Broker bilden, der dann über kein eigenes Retrievalsystem verfügt, sondern auf die Ergebnisse der Suchdienste zugreift. Die Inhalte und Metainformationen von Datenbanken können in einem automatischen Prozeß in das *SOIF* transformiert und an das Retrievalsystem weitergeleitet werden, so daß Zugriffe auf den Inhalt (z.B. durch generierte Suchformulare) denkbar sind.

3.3 MeDoc-IVS

Im Rahmen des Projektes M$_{\text{E}}$DOC[7] (**M**ultimedia **e**lectronic **Doc**uments), das sich zum Ziel setzt den Informationsaustausch und die Literaturversorgung in der Wissenschaft effizienter und effektiver zu gestalten, wurde im Teilprojekt 3 ein Informationsvermittlungssystem[8] (IVS) entwickelt. Dieses soll dem Informationssuchenden die Suche in verteilten und heterogenen Informationsquellen unter einer einheitlichen Oberfläche ermöglichen. Zu den Projektpartnern gehören u.a. die Gesellschaft für Informatik (GI), das Fachinformationszentrum Karlsruhe(FIZ) sowie der Springer-Verlag.

3.3.1 Architektur

Die in [BDG+96] eingeführte Architektur (siehe Abbildung 3.3) des Informationsvermittlungssystems unterteilt sich in drei Schichten:

- **Nutzeranbindungsschicht.** Die Nutzeranbindungsschicht stellt die Schnittstelle zwischen dem Nutzer und dem IVS zur Verfügung. Zu ihren Aufgaben gehören die *Transformation* (Umwandlung der Ein- und Ausgaben, Bereitstellung der Eingabemasken, Analyse der Eingabe und die Ausgabe der Ergebnisse), das *Weiterleiten von Aufträgen* (an die Komponenten der Vermittlungs- und Anbieteranbindungsschicht), die *Benutzerverwaltung*, die *Benutzerprofilverwaltung*, die *Ergebnismengenverwaltung*, das *Einbinden lokaler Datenbasen* (zur Speicherung der Ergebnisse und Volltexte) und die *Annotation* (Kommentare zu Ergebnissen und Volltexten).

[7] `http://medoc.informatik.tu-muenchen.de/`
[8] `http://hermes.offis.uni-oldenburg.de/~ua/`

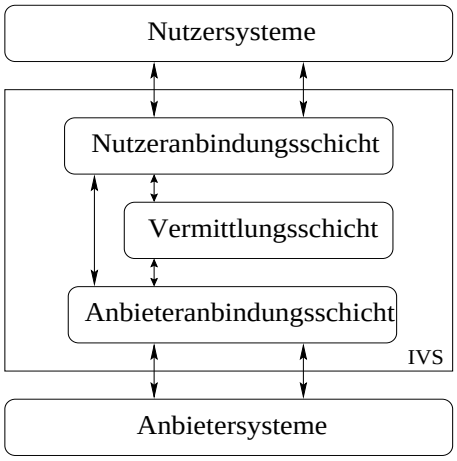

Abbildung 3.3: Schichtenarchitektur des Systems

- **Vermittlungsschicht.** Die Vermittlungsschicht ist für das *Verwalten von Anbieterbeschreibungen*, die *Auswahl geeigneter Anbietersysteme* und die *Rückgabe der ermittelten Anbietersysteme* zuständig. Anhand der Anbieterbeschreibungen wird für jede Anfrage eine Menge von Anbietersystemen ermittelt, die dem Nutzer als mögliche Datenbasis für die Suche dienen.

- **Anbieteranbindungsschicht.** Zu den Aufgaben der Anbieteranbindungsschicht gehört die *Transformation* der Nutzeranfragen (aus dem MEDOC-Protokoll und aus dem MEDOC-Schema auf die Schnittstellen der Anbietersysteme), das Weiterleiten der transformierten Anfragen an die Anbieter sowie die Rückgabe der Anfrageergebnisse. Um für Anfragen einen geeigneten Anbieter auswählen zu können, benötigt die Vermittlungsschicht *Anbieterbeschreibungen*, die inhaltliche und formale Aspekte der Anbieter umfassen.

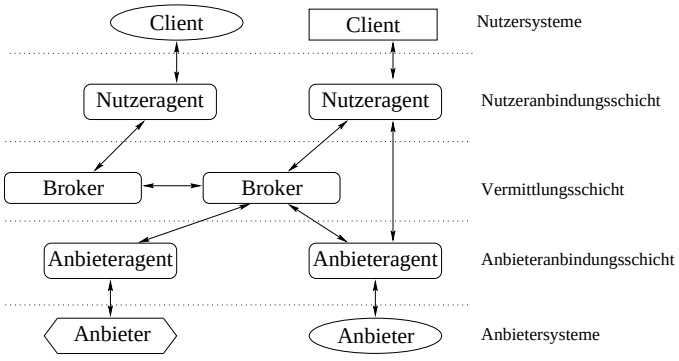

Abbildung 3.4: Komponenten des IVS

Jede der oben aufgeführten Schichten enthält Komponenten (Abbildung 3.4), die die schichtspezifischen Funktionen realisieren. Die Komponenten der Nutzeranbindungsschicht werden als *Nutzeragent/User-Agent (UA)*, die der Vermittlungsschicht als *Broker* und die der Anbieteranbindungsschicht als *Anbieteragent/ Provider-Agent (PA)* bezeichnet. Die Kommunikation zwischen den Schichten erfolgt nach dem M$_{\text{E}}$DOC-Protokoll.

3.3.2 Anfragebearbeitung

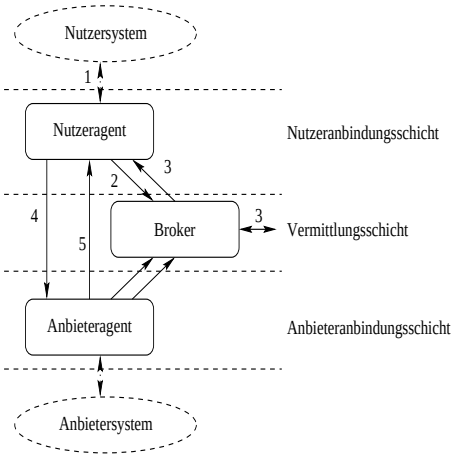

Abbildung 3.5: Datenfluß im IVS

Die Kommunikation der Nutzer mit dem M$_{\text{E}}$DOC-Dienst erfolgt mit Hilfe von WWW-Clienten. Die Bearbeitung einer Nutzeranfrage erfolgt in den folgenden Schritten (Abbildung 3.5):

1. Der Nutzer stellt eine Anfrage und kann auswählen, an welchen Broker oder welche Anbietersysteme diese geschickt werden soll.

2. Die Anfrage wird vom Nutzeragenten transformiert und an (genau) einen Broker bzw. an die selektierten Anbietersysteme weitergeleitet (in diesem Fall entfallen Schritt 3 und 4).

3. Der Broker ermittelt in Kooperation mit anderen Brokern[9] anhand der vorhandenen Anbieterbeschreibungen geeignete Anbietersysteme und liefert diese an den Nutzeragenten zurück. Diese Ergebnisliste kann vom Anwender modifiziert werden.

4. Nun leitet der Nutzeragent die Anfrage über die Anbieteragenten an die in der Ergebnisliste vorkommendenden Anbietersysteme weiter.

[9]In der zur Zeit eingesetzten Implementierung findet *keine* Kooperation der Broker statt.

5. Die Anbieteragenten ermitteln die Dokumente, die die Anfrage erfüllen, und liefern die Ergebnisse an den Nutzeragenten zurück. Hier werden sie gemischt[10] und anschließend in ein für das Nutzersystem lesbares Format transformiert und dem Nutzer präsentiert.

3.3.3 Kriterien

Datenbasis-Auswahl. Der Nutzeragent bietet dem Anwender folgende Möglichkeiten:

- Suche über Brokern, wobei die in der Ergebnisliste enthaltenen Anbietersysteme
  - automatisch
  - nach Überarbeitung durch den Nutzer

  kontaktiert werden.
- Suche direkt in der Datenbasis des Anbieters.

Bewertung: ⊕

Heterogenität der Retrievalsysteme. Durch die in der Schichtenarchitektur definierte Anbieteranbindungsschicht kann im Prinzip jedes Retrievalsystem verwendet werden. Das MEDOC-System ist zur Zeit mit Volltextspeichern, WAIS-, NCSTRL-, Sybase-Datenbanken sowie diversen Systemen, die via Z39.50 Gateway erreichbar sind, verbunden.

Bewertung: ⊕

Integration existierender Suchdienste. Die Anbindung von Suchdiensten ist prinzipiell möglich. Es kann jede Datenbank als Datenbasis verwendet werden, sofern die Anfragesprache und Rückgabeformate bekannt sind. Die für die Transformation notwendigen Metainformationen[11] müssen dem Anbieteragenten zur Verfügung stehen. Als Beispiel ist hier *ARIADNE*[12] [DLSZ98] zu nennen.

Bewertung: ⊕

Integration von Datenbanken. Die an das IVS angeschlossenen Datenbasen umfassen keine Dokumente, die dynamisch erzeugt werden.

Bewertung: ⊘

[10] Das Mischen erfolgt u.a. aus Performancegründen nicht vom Broker, obwohl dies Aufgabe der Vermittlungsschicht wäre. In der zur Zeit verfügbaren Implementierung werden die Anfrageergebnisse *nicht* gemischt.

[11] In SQL-Datenbanken sind dies u.a. Schlüsselworte und Attributnamen.

[12] `http://ariadne.inf.fu-berlin.de:8000/index.html`

Metadatenhaltung. Wenn ein Anbietersystem in den MEDOC-Dienst aufgenommen werden will, muß ein Anbieteragent erzeugt werden. Der Agent benötigt Informationen über die Art und Weise der Anmeldung sowie die Sprache, in der er mit dem Anbietersystem kommunizieren kann. Der Anbieter stellt weiterhin Metadaten bereit, die zum einen inhaltliche Informationen[13] und zum anderen organisatorische Informationen[14] enthalten. Die für die Anbieterauswahl relevanten Metadaten werden an diejenigen Broker weitergeleitet, denen der Anbieteragent bekannt ist. Die Gewinnung und die Aktualisierung der Metadaten erfolgt manuell durch den MEDOC-Systembetreuer oder einen Betreuer auf Anbieterseite.

Ranking. Für das Ranking sind die jeweiligen Anbietersysteme zuständig; das Mischen der Ergebnisse sollte laut Konzeption im Nutzeragenten erfolgen. Aus Zeitgründen wurde in der aktuellen Implementierung das Mischen der Teilergebnisse nicht umgesetzt. Deshalb erfolgt lediglich eine Auflistung aller Anbietersysteme mit ihren Anfrageergebnissen.

Bewertung: $\oplus$ / $\ominus$

Verteiltes Retrieval. Jeder Nutzeragent verfügt über eine Tabelle aller Broker, die von Hand gewartet wird. Dadurch kann der Nutzeragent im Falle, daß 'sein' Broker nicht verfügbar ist, die Anfrage an einen der anderen Broker stellen. Eine Kooperation der Broker zur Beantwortung der Anfrage findet ist der derzeitigen Implementierung nicht statt. Die Suchanfrage wird an alle selektierten/ermittelten Anbieteragenten weitergeleitet, die ihrerseits das zugeordnete Anbietersystem befragen. Es findet also ein verteiltes Retrieval statt.

Bewertung: $\oplus$

Verteiltheit/Replikation. Die Daten über die Anbietersysteme werden im zugeordneten Anbieteragenten und in den Brokern gehalten, wobei jeweils nur die notwendigen Informationen gespeichert werden (siehe Metadatenhaltung). Da die Datenbestände der Broker relativ statisch sind und sich im ersten Prototyp gezeigt hat, daß ein Broker als einzige zentrale Komponente die Schwachstelle bildet, wurde im erweiterten Prototyp die Replikation der Broker [Dre97] in Betracht gezogen. Zwischen den Replikaten wurde nur schwache Konsistenz gefordert, da Änderungen in der Anbieterbeschreibung häufig keine grundlegende andere Bewertung des Anbieters nach sich ziehen und Neuanmeldungen von Anbietersystemen nicht unbedingt sofort jedem Broker bekannt sein müssen. In der Praxis ist bis heute lediglich eine manuelle Replikation möglich.

Bewertung: $\oplus$

[13] Welche Daten sind beim Anbieter gespeichert ?

[14] durchschn. Zugriffsdauer, Kosten, Login, Paßwort, DB-Schema

Erweiterbarkeit/Verfügbarkeit. Das MEDOC-IVS ist ein offenes System; die Integration von neuen PAs (und somit Anbietersystemen) wurde beispielsweise relativ einfach gehalten. Bei der Implementierung des Protokolls wurde auf Einfachheit, Stabilität und Erweiterbarkeit Wert gelegt [DLM98]. Die Konzeption des IVS umfaßt einige Aspekte, die für den in dieser Arbeit zu konzipierenden Suchdienst nicht von Belang sind. So spielt die Kostenkontrolle (Zugriff auf gebührenpflichtige Dienste), die durch eine zweistufige Anfragebearbeitung gewährleistet wird, keine Rolle. Die Anbindung von Volltextsystemen [MA98] geht ebenfalls weit über die hier benötigte Funktionalität hinaus. Weiterhin ist der Prozeß der Indexbearbeitung (Sammeln der Dokumente, Einspeisen der extrahierten Informationen in das Retrievalsystem) nicht näher beschrieben, da dies Aufgabe des Anbieters ist. Als überaus gut erweist sich die (teilweise) recht umfangreiche Literatur zur Konzeption des IVS und deren Komponenten. Aus dem Bereich der Implementierung gibt es leider kaum frei verfügbare Unterlagen; der Quellcode selbst ist nicht frei verfügbar. Deshalb kann keine der Komponenten des Systems für die eigene Realisierung verwendet werden. Trotz der vielen positiven Aspekte kann es deshalb keine Empfehlung als Ausgangsbasis für die Eigenentwicklung geben.

Bewertung: ⊗

3.3.4 Fazit

Das MEDOC-IVS bietet einige gute Konzepte, die im später vorzustellenden Realisierungsvorschlag berücksichtigt werden. Die zweistufige Anfragebearbeitung hingegen scheint auf die eigene Konzeption nicht übertragbar, da keine Kostenkontrolle stattfinden muß.[15] Ein Zwischenspeichern der Anfrageergebnisse auf Seiten des UA scheint sinnvoll zu sein, da der Broker entlastet wird und eine weitere Nutzung der Anfrageergebnisse erfolgen kann. Zusammenfassend läßt sich sagen, daß das MEDOC-IVS aufgrund der etwas anderen Anforderungen an das Design keine gute Basis für den eigenen Realisierungsvorschlag bildet.

Anmerkung: Die Technologie und Infrastruktur des MEDOC-Dienstes werden durch InterDoc[16] in das Global-Info Projekt[17] eingebracht und durch Nutzer evaluiert und mit anderen Werkzeugen verglichen. Im Rahmen von InterDoc werden einige Änderungen am IVS vorgenommen - das bisher fehlende globale Ranking

[15] Eine wichtige Rolle im Design des MEDOC-IVS spielt das Angebot von kostenpflichtigen Dokumenten und Recherchemöglichkeiten durch diverse Verlage wie Heise und Springer. Deshalb muß dem Nutzer die Möglichkeit gegeben werden, den Zugriff auf kommerzielle Angebote zu überwachen und, falls gewünscht, zu unterbinden. Dies ist im hier zu entwerfenden Suchdienst nicht erforderlich.

[16] Interdisziplinäre Dokumentenverarbeitung auf Basis des MEDOC-Dienstes [int98]

[17] Globale Elektronische und Multimediale Informationssysteme für Naturwissenschaft und Technik [glo]

wird aber auch im neuen IVS nur ansatzweise implementiert.

3.4 freeWAIS

Der durch die Firmen *Thinking Machines*, *Apple*, *KPMG Peat Marwick* und *Dow Jones Information Service* entwickelte Volltextdatenbank-Suchdienst *WAIS* (Wide Area Information Servers) wurde im Jahr 1991 eingeführt. Anbieter stellen in ihren Datenbanken, die sich meistens auf ein Themengebiet beschränken (bspw. Sport, Biologie, Astronomie), eine Sammlung von Dokumenten zur Verfügung, die vom Benutzer, damals via telnet (swais) oder Email bzw. speziellen Clients (XWAIS), heute mittels WWW-Interface (*SFgate* [Gö98]) durchsucht werden können. Aufgrund der wachsenden Popularität des Systems entschloß man sich, WAIS zu kommerzialisieren. Die letzte herausgegebene Public Domain-Version (*wais-8-b5*) wurde vom *Clearinghouse for Networked Information Discovery and Retrieval* (CNIDR) übernommen und fortan unter dem Namen *freeWAIS* weiterentwickelt. Während das CNIDR das anfänglich verwendete Z39.50-88 Protokoll an die neuere Version Z30.50-92 anpaßte [Pfe95b], kam es andernorts zu einer Reihe von Weiterentwicklungen, die aus Zeitgründen nicht in *freeWAIS* integriert wurden. Infolgedessen entwickelten sich eine Reihe von Ablegern. Ein solcher Ableger ist das an der Universität Dortmund entstandene *freeWAIS-sf* [PFH95], das zusammen mit dem ebenfalls an der Universität Dortmund entwickelten WWW-Interface *SFgate* eine weite Verbreitung gefunden hat.

3.4.1 Architektur

FreeWAIS basiert auf einer Client-Server Architektur (Abbildung 3.6), bei der Client und Server mittels des WAIS-Protokolles miteinander kommunizieren. Die Clients verfügen über *source descriptions* (Rechnername, IP-Adresse, Serverport, Datenbankname, Beschreibung des Datenbankinhaltes), die die verfügbaren Datenbanken und die dazugehörigen Server beschreiben. Spezielle Server, die das sogenannte *directory of servers* (Serververzeichnis[18]) verwalten, ermöglichen dem Nutzer die Suche in den verfügbaren Datenbanken. Somit kann die Suche trotz des umfangreichen Angebotes an Datenquellen auf einige wenige vielversprechende Server eingeschränkt werden. Ein Server greift für die Beantwortung der Anfragen auf die WAIS-Datenbank, die aus den erfaßten Dokumenten und dem daraus gebildeten Index besteht, zu.

3.4.2 Anfragebearbeitung

1. Mit Hilfe des "directory of servers" werden die Server ermittelt, die Dokumente enthalten könnten, die der Suchanfrage entsprechen. Dieser Schritt

[18] Eine Datenbank in der die verfügbaren WAIS-Server gespeichert sind.

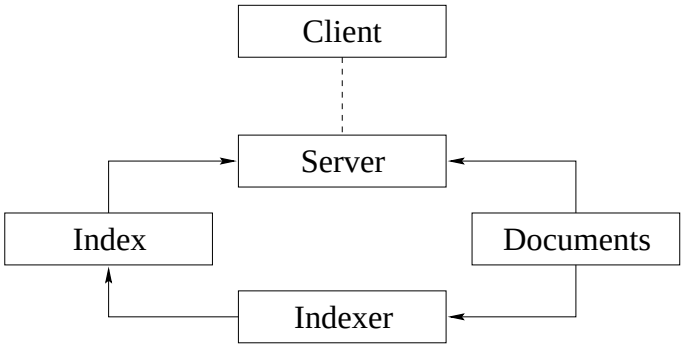

Abbildung 3.6: freeWAIS Architektur

ist optional.

2. Die Anfrage wird an die selektierten Server weitergeleitet und die Anfrageergebnisse bestimmt.

3. Die zurückgegebenen Ergebnisse werden gerankt, gemischt und dem Nutzer präsentiert.

3.4.3 Kriterien

Datenbasis-Auswahl. Der Anwender kann aus den verfügbaren WAIS-Datenbanken beliebig viele auswählen, an die anschließend die Anfrage weitergeleitet wird.

Bewertung: $\oplus$

Heterogenität der Retrievalsysteme. Als Retrievalsystem können lediglich WAIS-Datenbanken Verwendung finden.

Bewertung: $\oslash$

Integration existierender Suchdienste. Es können keine anderen Suchdienste angesprochen werden; lediglich WAIS-Datenbanken können befragt werden.

Bewertung: $\oslash$

Integration von Datenbanken. Die Integration von Datenbanken, d.h. dynamischen Dokumenten, ist nicht möglich.

Bewertung: $\oslash$

Metadatenhaltung. Zu jeder WAIS-Datenbank existiert eine Beschreibung [Pfe95a], die Auskunft über die in dieser Datenbank vorhandenen Dokumente gibt. Der Indexierer erzeugt automatisch eine Keyword-Liste, die, zusammen mit manuell erfaßten Kommentaren, vom “directory of servers“ zur Ermittlung von Datenbasen genutzt werden können.

Ranking. Das Ranking von WAIS basiert auf dem Vektorraummodell. Wenn die Suche in mehreren Datenbanken erfolgt, werden die Ergebnisse gemischt und in einer einheitlichen Form präsentiert.

Bewertung: $\oplus$ / $\oplus$

Verteiltes Retrieval. Die Anfragebearbeitung erfolgt parallel in den kontaktierten Retrievalsystemen (WAIS-Datenbanken).

Bewertung: $\oplus$

Verteiltheit/Replikation. Der Index wird zusammen mit den indexierten Dokumenten lokal gehalten. Lediglich die *source descriptions* (Rechnername, IP-Adresse, Serverport, Datenbankname, Beschreibung des Datenbankinhaltes), die die verfügbaren Datenbanken beschreiben, können in mehreren *directories of servers* vorkommen. Eine Replikation der Datenbestände findet nicht statt.

Bewertung: $\odot$

Erweiterbarkeit/Verfügbarkeit. Die Einbindung neuer Dokument-Typen in die Datenbank erfolgt durch eine Beschreibung des Dokumentformates in den dafür vorgesehenen Konfigurationsdateien; somit können beliebige Dokumente indexiert werden. Eine Anbindung von Datenbasen, die nicht auf WAIS basieren, scheint nicht möglich zu sein. Da auch keine offene Schnittstelle (wie z.B. bei Harvest) zwischen Gatherer und Retrievalsystem (WAIS-Datenbank) definiert wurde, ist die gesamte Software umzuschreiben, um heterogene Retrievalsysteme zu integrieren. Insgesamt betrachtet ist freeWAIS kein System, das bezüglich der nicht erfüllten Kriterien gut erweiterbar erscheint. Das Kriterium der Verfügbarkeit hingegegen ist erfüllt. Der Programmcode ist neben diversen Dokumentationen wie FAQs und Handbüchern im WWW frei verfügbar. Leider beschränken sich die Dokumentationen größtenteils auf Installation und Konfiguration der Komponenten. Eine Newsgruppe[19] komplettiert das Angebot an Informationen.

Bewertung: $\otimes$

3.4.4 Fazit

Der Suchdienst freeWAIS offenbart einige Schwächen. Durch die Beschränkung, ausschließlich WAIS/freeWAIS Datenbanken als Datenbasis zu verwenden, ist die Heterogenität der Retrievalsysteme sehr eingeschränkt. Da der Indexierer lediglich lokal gespeicherte Dokumente verarbeiten kann, ist es notwendig, für jede nicht-lokale Datenbasis einen eigenen Index zu erstellen. Der große Speicherplatzbedarf der Indizes bietet einen weiteren Ansatz zur Kritik. Das Konzept des "directory of servers" hingegen ist als positiv zu bewerten, da die Selektion der

[19] `news:comp.infosystems.wais`

vielversprechendsten Datenbasen vereinfacht wird und entscheidend zur Güte der Anfrageergebnisse beiträgt. Insgesamt betrachtet ist auch freeWAIS keine geeignete Basis für die eigene Realisierung einer verteilten Suchmaschine.

3.5 Z39.50

Die ANSI/NISO Z39.50-1995 *Information Retrieval Application Service Definition and Protocol Specification* [InR95] ist die neueste Version der ursprünglich im Jahre 1984 von der National Information Standards Organization (NISO) eingeführten Spezifikation. Die Weiterentwicklung übernahm die 1989 eigens gegründete Z39.50 Maintenance Agency[20]. Stand anfangs die bibliographische Nutzung im Vordergrund, so geht heute es um den Zugriff auf Informationen aller Art (Bilder, Texte, Finanzdaten, ...). Da viele Hersteller, Informationsanbieter, Consultants, Händler aber auch Universitäten großes Interesse zeigten, wurde 1990 die Z39.50 Implementors Group[21] (ZIG) gegründet, die sich um die praktische Umsetzung kümmert. Zu den Mitgliedern der ZIG zählen unter anderem das Fachinformationszentrum (FIZ) Karlsruhe und das Massachusetts Institute of Technology (MIT).

3.5.1 Architektur

Die Protokollspezifikation definiert das Format und die Prozeduren, die dem Nachrichtenaustausch zwischen Client und Server dienen, und wird in dieser Arbeit nicht weiter betrachtet. Die *Information Retrieval Service Definition* beschreibt die Vorgänge, die zwischen einer anfragenden Applikation (Client) und der antwortenden Applikation (Server) ablaufen. Die in Abbildung 3.7 dargestellten Facilities (Einrichtungen), bei denen es sich um logische Zusammenfassungen von Services (Diensten) handelt, werden nun näher beschrieben:

Initialization Facility. Mittels *Init Service* wird eine Verbindung zwischen Client und Server aufgebaut, nachdem die zum Verbindungsaufbau erforderlichen Parameter ausgetauscht wurden.

Search Facility. Auch diese Einrichtung bietet nur einen Dienst an: den *Search Service*. Der Suchdienst ermöglicht es dem Client Anfragen an Datenbanken zu stellen, die mit dem Server verbunden sind, und Informationen über die Ergebnisse abzurufen.

Retrieval Facility. Mit Hilfe des *Present Service* kann der Client die Anfrageergebnisse (auch Teile der Ergebnismenge) vom Server anfordern. Sollten die angeforderten Ergebnisse nicht in ein (in der Größe vorher festgelegtes)

[20] `http://lcweb.loc.gov/z3950/agency/`
[21] `http://lcweb.loc.gov/z3950/agency/zig/zig.html`

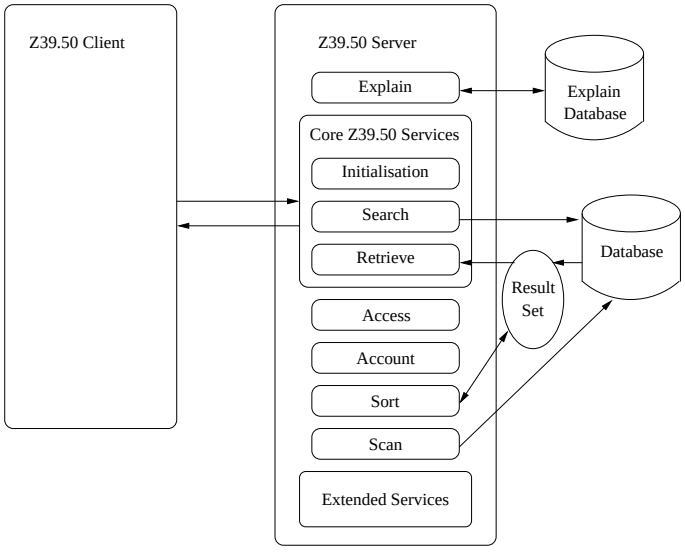

Abbildung 3.7: Z39.50 Facility Diagram

Antworttupel passen, so kann mit Hilfe des *Segment Service* die Ergebnismenge in mehreren Segmenten übertragen werden.

Result-set-delete Facility. Durch den *Delete Service* kann der Client den Server veranlassen, auf dem Server gespeicherte Ergebnismengen zu löschen.

Access Control Facility. Der *Access-control Service* kann vom Server zur Autorisierung des Clients (Paßwort) aber auch zur Verschlüsselung (z.B. durch public-key Verschlüsselungsverfahren) genutzt werden.

Accounting/Resource Control Facility. Drei Dienste stellt die Accounting/ Resource Control Facility zur Verfügung: der *Resource-control Service* ermöglicht es dem Server, Nachrichten an den Client zu schicken (um ihn auf Probleme oder auf Fortschritte bei der Abarbeitung des Auftrages aufmerksam zu machen), der *Trigger-resource-control Service* erlaubt es dem Client, Nachrichten an den Server zu schicken (Anforderung von mehr Ressourcen, Statusbericht, Abbruch der Operation) und der *Resource-report Service* dient dem Client dazu, vom Server Statusberichte anzufordern. Letzterer Dienst ist nicht identisch mit dem *Trigger-resource-control Service*, da er vom Server bestätigt werden muß (was im ersteren Fall nicht nötig war).

Sort Facility. Nachdem vom Server die Anfrageergebnismengen ermittelt hat, können diese durch den *Sort Service* gemischt und sortiert werden. Dazu sendet der Client eine Anfrage, in der er dem Server mitteilt, welche Ergebnismengen diesbezüglich aufzubereiten zu sind.

Browse Facility. Der *Scan Service* bietet dem Client die Möglichkeit durch geordnete Listen (bspw. Titel oder Autor) zu browsen.

Extended Services Facility. Der *Extended Services Service* dient dazu, auf dem Server Tasks auszuführen, die zwar Bestandteil der Informationsermittlung sind aber nicht zu den Z39.50 Diensten gehören. Ein solcher Task könnte beispielsweise das Exportieren eines Dokumentes sein oder auch das Update einer Datenbank.

Explain Facility. Diese Einrichtung ist die einzige, die keine eigenen Dienste anbietet, sondern die Dienste der Search und Retrieval Facility nutzt. Sie ermöglicht dem Client Informationen über den Server zu erhalten (z.B. verfügbare Datenbanken).

Termination Facility. Die Aufgabe des *Close Service* besteht darin, aktive Operationen abzubrechen und die Verbindung zwischen Client und Server zu schließen. Dieser Dienst kann vom Client und vom Server aufgerufen werden.

Einen sehr guten Überblick von Z39.50 im Allgemeinen [Zo98] sowie die technischen Details [Ztd98] bietet die *Biblio Tech Review*[22].

3.5.2 Anfragebearbeitung

1. Der Client schickt eine Suchanfrage an den Server, wobei folgende Parameter angegeben werden können:
   - Namen der zu durchsuchenden Datenbanken
   - ob die Ergebnistupel Teil der Antwort sein sollen
   - die Syntax, in der die Ergebnistupel erwartet werden
   - der Name, unter dem die Ergebnismenge gespeichert wird, um darauf später zuzugreifen
2. Der Server liefert die Anzahl der Ergebnistupel sowie (falls gefordert) die Tupel selbst zurück.

3.5.3 Kriterien

Datenbasis-Auswahl. Der Client spezifiziert in seiner Suchanfrage, an welche Datenbanken die Anfrage geschickt wird. Durch die Nutzung der *Explain Facility* kann der Client Informationen, die zur Ermittlung der zu befragenden Datenbanken nötig sind, erhalten. *Bewertung:* $\oplus$

[22] `http://www.biblio-tech.com/`

Heterogenität der Retrievalsysteme. Die Definition eines abstrakten Modelles zur Beschreibung von Datenbanken ermöglicht den Einsatz von beliebigen Datenbanken als Retrievalsystem. Zu jedem konkreten Datenbanksystem ist demzufolge nur eine Abbildung vom abstrakten Modell auf die von der Datenbank verwendete Implementierung anzugeben.

Bewertung: $\oplus$

Integration existierender Suchdienste. Der Z39.50 Standard dient zur Spezifikation von Strukturen und Prozeduren, die es dem Client ermöglichen Anfragen an Datenbanken, die durch einen Server bereitgestellt werden, zu stellen. Die Integration von Suchdiensten beschränkt sich demzufolge auf beliebige Datenbanken, die befragt werden können.

Bewertung: $\oslash$

Integration von Datenbanken. Da der Z39.50 Standard primär für den Bereich des Bibliothekswesens und der Informationsvermittlungssysteme konzipiert wurde und sich die Angebote auf statische Dokumente beschränken, ist eine Integration von dynamischen Dokumenten nicht berücksichtigt.

Bewertung: $\oslash$

Metadatenhaltung. Die *Explain Facility* stellt unter Nutzung der *Explain Database* umfangreiche Informationen über die verfügbaren Datenbanken, Schemainformationen, Attributmengen und weiteren Spezifikationen bereit. Der Standard definiert lediglich die Metainformationen, die durch den Server in der *Explain Database* bereitgestellt werden sollten, und nicht die Art der Gewinnung und des Updates.

Ranking. Ein Ranking gibt es nicht. Die Anfrageergebnisse (*result sets*) werden mit Hilfe der *Sort Facility* sortiert und im Fall, daß mehrere Datenbanken befragt wurden, auch gemischt.

Bewertung: $\oslash$ / $\oslash$

Verteiltes Retrieval. Der Client kann gleichzeitig an alle auf dem kontaktierten Server verfügbaren Datenbanken Anfragen stellen. Ob die Bearbeitung der Anfrage in den Datenbanken parallel oder sequentiell erfolgt, wird in der Initialisierungsphase festgelegt.

Bewertung: $\odot$

Verteiltheit/Replikation. Die *Explain Facility* umfaßt alle Informationen zu den ansprechbaren Datenbanken. Im Z39.50 Standard werden keine weiteren Festlegungen bezüglich Replikation und Verteiltheit des Systems getroffen.

Bewertung: $\oslash$

Erweiterbarkeit/Verfügbarkeit. Die Betrachtung der Z39.50 Spezifikation läßt keine Bewertung der Verfügbarkeit zu. Dazu wäre die Betrachtung einer konkreten Implementierung nötig, was aus Zeitgründen aber nicht erfolgt ist. Durch die *Extended Services Facility* ist eine Erweiterung der Funktionalität des Suchdienstes möglich.

Bewertung: $\otimes$

3.5.4 Fazit

Z39.50 eignet sich nicht als Basis für den eigenen Suchdienst. Die Kommunikation zwischen Client und Server ist für die Anwendung in dieser Arbeit viel zu umfangreich und größtenteils nicht erforderlich. Da außerdem viele der geforderten Kriterien durch das System nicht erfüllt werden, die Umsetzung dieser Kriterien mit Z39.50 als Basis nicht sehr vielversprechend erscheint und keine konkrete Implementierung analysiert wurde, spielt das System in der weiteren Betrachtung des zu entwickelnden Suchdienstes keine Rolle.
Anmerkung: Unter dem Namen *ZDSR* (**Z**39.50 Profile for Simple **D**istributed **S**earch and Ranked **R**etrieval) [zds] wurde ein Profil[23] des Z39.50-1995 Standards entwickelt, der von STARTS[24] abstammt. In ZDSR werden Prozeduren spezifiziert, die die verteilte Suche und das gerankte Retrieval unterstützen. Dieses Profil wurde in die Betrachtung nicht einbezogen.

3.6 Zusammenfassung und Tabellarischer Vergleich

Die Analyse der in den Systemen verwendeten Konzepte zeigte einige Ansätze, die im eigenen Realisierungsvorschlag durchaus Verwendung finden können.

Im Bereich der Datenbasisauswahl bieten die betrachteten Systeme die manuelle und die automatische Selektion an. Am günstigsten erscheint eine hybride Vorgehensweise; dem Nutzer sollte die Möglichkeit gegeben sein, die Auswahl von vornherein zu beschränken. Dieses bietet sich insbesondere dann an, wenn Datenbasen ein schlechtes Antwortzeitverhalten zeigen bzw. wenn der 'Suchraum' eingeschränkt werden soll. Das im MEDOC-IVS verwendete Konzept der Kooperation der Broker ist dem von freeWAIS (ein "directory of servers" ermit-

[23] Ein Profil spezifiziert die Verwendung eines bestimmten Standards zur Unterstützung einer Anwendung (z.B. WAIS), Funktion (z.B. Suche nach Titeln), Gemeinschaft (z.B. Mediziner, Musiker) oder Umgebung (z.B. das Internet). Mit "Verwendung eines Standards" ist hier die Auswahl (die im Standard nicht festgelegt ist) von Optionen, Teilmengen und Parameterwerten gemeint.

[24] STARTS steht für "Stanford Protocol Proposal for Internet Retrieval and Search". In diesem Projekt wurden Voraussetzungen für die verteilte Suche und das gerankte Retrieval entwickelt. [GCGM+97]

telt die erfolgversprechendsten Datenbanken) vorzuziehen, da sich eine zentrale Komponente als Flaschenhals erweisen kann. Eine eventuelle Replikation dieser Komponente könnte dem entgegenwirken.

Bis auf freeWAIS gewährleisten alle Vertreter durch ihre Konzepte (Harvest: Broker/Indexer Interface, M$_{\text{E}}$DOC-IVS: Anbieteragent, Z39.50: Abstraktes Modell zur Beschreibung von Datenbanken) die Verwendung von heterogenen Retrievalsystemen.

Die Integration von Suchdiensten ist nur beim M$_{\text{E}}$DOC-IVS möglich. Ein Übersetzungsmodul sorgt für die Umsetzung der der Anfrage in die entsprechende Zielsprache und die Transformation der Ergebnisse. Eine Umsetzung dieses Konzeptes scheint auch für das Harvest-System möglich zu sein, wobei die Funktionalität des Index/Search Subsystems durch den externen Suchdienst realisiert wird und der Broker somit über keine eigenständige Datenbasis (die durch Gatherer geupdated wird) mehr verfügt.

Keiner der Vertreter unterstützt direkt die Indexierung von dynamischen Dokumenten. Deshalb müssen eigene Konzepte ausgearbeitet werden, die sich in die bestehenden Architekturen einpassen lassen.

Mit Ausnahme von Z39.50 bieten alle Systeme ein Ranking der Anfrageergebnisse an; das Retrievalsystem bestimmt die Art des Rankings. Das globale Ranking von Anfrageergebnissen erfolgt lediglich in freeWAIS. Da die Broker im Harvest System nur die lokale Datenbasis befragen, ist ein Mischen von Anfrageergebnissen nicht nötig. Im M$_{\text{E}}$DOC-IVS erfolgte aus Zeitgründen keine Implementierung des Mischens.

Das M$_{\text{E}}$DOC-IVS und freeWAIS unterstützen die Selektion der Datenbasen und gewährleisten eine parallele Bearbeitung der Anfragen. In Z39.50 kann weiterhin angegeben werden, ob die Bearbeitung parallel oder sequentiell erfolgen soll. Der Suchdienst Harvest bietet kein verteiltes Retrieval an.

Die Systeme liegen jeweils verteilt vor.[25] Harvest und das M$_{\text{E}}$DOC-IVS bieten darüber hinaus die Replikation der Broker (im Falle von Harvest inklusive der dazugehörigen Datenbasis) zur besseren Lastverteilung und Verfügbarkeit an. In Z39.50 und freeWAIS werden keine Aussagen bezüglich der Replikation getroffen.

Hinsichtlich ihrer praktischen Verwendbarkeit als Ausgangsbasis für den eigenen Realisierungsvorschlag einer verteilten Suchmaschine scheiden das M$_{\text{E}}$DOC-IVS, freeWAIS und Z39.50 aus. Ersteres ist kein frei verfügbares System und weist aufgrund anderer Design-Anforderungen an das System einige Konzepte auf, die hier keine Verwendung finden können. FreeWAIS mangelt es an der Erweiterbarkeit und die betrachtete Z39.50-Spezifikation bietet in ihrer hier betrachteten Form keine praktische Verwendbarkeit.

In Tabelle 3.1 sind die Kriterien und ihre Bewertung in den entsprechenden Systemen überblicksweise aufgelistet. Anhand dieser Tabelle lassen sich offene

[25] In Z39.50 wird lediglich eine Client-Server Architektur eingeführt; über eine physische Verteilung des Systems gibt es keine Angaben.

Probleme, die in der Konzeption zu berücksichtigen sind, ablesen. Der abschließende Abschnitt listet diese Probleme auf.

3.6.1 Schlußfolgerungen

Zusammenfassend betrachtet liefert kein System für alle Kriterien ein Konzept und realisiert es auch. Durch seine gute Bewertung im Kriterium Erweiterbarkeit/Verfügbarkeit soll das Harvest-System die Grundlage für die sich anschließende Konzeption bilden. Eine Betrachtung der tabellarischen Übersicht zeigt, daß folgende Probleme zu bearbeiten sind:

- Datenbasisauswahl - Kooperation der Broker (Entscheidung für einen zentralen Ansatz wie freeWAIS ihn bietet oder für einen dezentralen Ansatz wie er im M_EDOC-IVS in Betracht gezogen wurde)
- Integration von Suchdiensten (Anbindung eines Suchdienstes an den Broker als "Retrievalsystem")
- Integration von Datenbanken (Nutzung der Funktionalität des Gatherers zur Erfassung der Daten und Metadaten einer Datenbank)
- globales Ranking (Welche Informationen sind notwendig, um ein globales Ranking zu realisieren ?)
- verteiltes Retrieval (Wie muß die bestehende Harvest-Architektur modifiziert werden, um ein verteiltes Retrieval zu realisieren ?)

Das nächste Kapitel wird versuchen diese Probleme zu lösen, wobei der Umfang der Betrachtungen relativ beschränkt sein wird, da einige dieser Probleme von sehr komplexer Natur und Bestandteil aktueller Forschungen sind. Deshalb wird gegebenenfalls auf die entsprechende Literatur verwiesen.

Kriterium	Harvest	MEDOC-IVS	freeWAIS	Z39.50
Datenbasisauswahl	$\oslash$	$\oplus$	$\oplus$	$\oplus$
Heterogenität der Retrievalsysteme	$\oplus$	$\oplus$	$\oslash$	$\oplus$
Integration existierender Suchdienste	$\oslash$	$\oplus$	$\oslash$	$\oslash$
Integration von Datenbanken	$\oslash$	$\oslash$	$\oslash$	$\oslash$
Metadatenhaltung[a]	-	-	-	-
Ranking (lokal/global)	$\oplus$ / $\oslash$	$\oplus$ / $\ominus$	$\oplus$ / $\oplus$	$\oslash$ / $\oslash$
Verteiltes Retrieval	$\oslash$	$\oplus$	$\oplus$	$\odot$
Verteiltheit/Replikation	$\oplus$	$\oplus$	$\odot$	$\oslash$
Erweiterbarkeit/ Verfügbarkeit[b]	$\bigoplus$	$\bigotimes$	$\bigotimes$	$\bigotimes$

Erläuterung der verwendeten Symbole:

$\oslash$ das Kriterium ist keine Anforderung an das System; es existiert demzufolge kein Konzept, das im eigenen Realisierungsvorschlag verwendet werden kann

$\ominus$ Kriterium nicht erfüllt; es gibt kein Konzept, das betrachtet werden kann

$\odot$ Kriterium erfüllt; das verwendete Konzept ist nicht für den eigenen Realisierungsvorschlag geeignet

$\oplus$ Kriterium erfüllt; das Konzept ist gut in den eigenen Vorschlag zu integrieren

$\bigotimes$ nicht verwendbar

$\bigoplus$ verwendbar

[a] keine Bewertung

[b] zusammengefaßt unter dem Gesichtspunkt, wie gut die praktische Einsetzbarkeit als „Basis“ für den eigenen Realisierungsvorschlag ist

Tabelle 3.1: die Kriterienbewertung im Überblick

Kapitel 4

Architekturvorschlag

In der Motivation zu dieser Studienarbeit (Abschnitt 1.5) wurde deutlich, daß eine verteilte Suchmaschine eine verteilte Architektur aufweisen und die verteilte Indexierung/Suche realisieren sollte. Ausgehend von einer allgemeinen Darlegung der Architektur werden in den folgenden Abschnitten die konkreten Komponenten des Systems vorgestellt und Realisierungsvorschläge für die Implementierung gegeben. Dabei fließen einige der im vorherigen Kapitel als positiv bewerteten Konzepte in die eigene Entwicklung ein. Bei der Konzeption ist ferner darauf zu achten, daß die konkrete Realisierung der Suchmaschine unter Umständen Einsatz im *Online-Dienst Mecklenburg-Vorpommern*[1] finden wird und somit einige der bereits bestehenden Strukturen zu nutzen sind. Aus diesem Grunde beginnt dieses Kapitel mit einer kurzen Einführung in die zur Zeit vorliegenden Strukturen dieses Dienstes.

4.1 Der Online-Dienst MV-Info

Der Informationsdienst, der vom Forum für Informations-Services in Mecklenburg-Vorpommern (ISMV) betrieben wird und für dessen Inhalt diverse Unternehmen und Forschungseinrichtungen des Landes verantwortlich sind, beruht auf einem Katalogsystem. Informationsanbieter der unterschiedlichsten Bereiche wie Politik, Bildung und Kultur können sich hier in den entsprechenden Kategorien registrieren lassen. Der zur Zeit verwendete Suchdienst Swing[2] (**S**uchdienst für **W**WW-basierte **I**nformationssysteme der **n**ächsten **G**eneration) basiert auf dem System Harvest, wobei 76 Domänen [Sd998] durch mehrere Gatherer indexiert und mit Hilfe eines Brokers durchsucht werden. Auf weitere Einzelheiten wird in den Abschnitten, die die einzelnen Komponenten beschreiben, näher eingegangen.

[1] `http://www.m-v.de/`
[2] `http://swing.informatik.uni-rostock.de/`

4.2 Die Architektur

Die in Abbildung 4.1 dargestellte Architektur des Suchdienstes führt mehrere Schichten ein, die jeweils spezifische Aufgaben übernehmen. Die Funktionalität einer Schicht wird durch eine oder mehrere Komponenten realisiert, die dieser Schicht zugeordnet ist/sind.

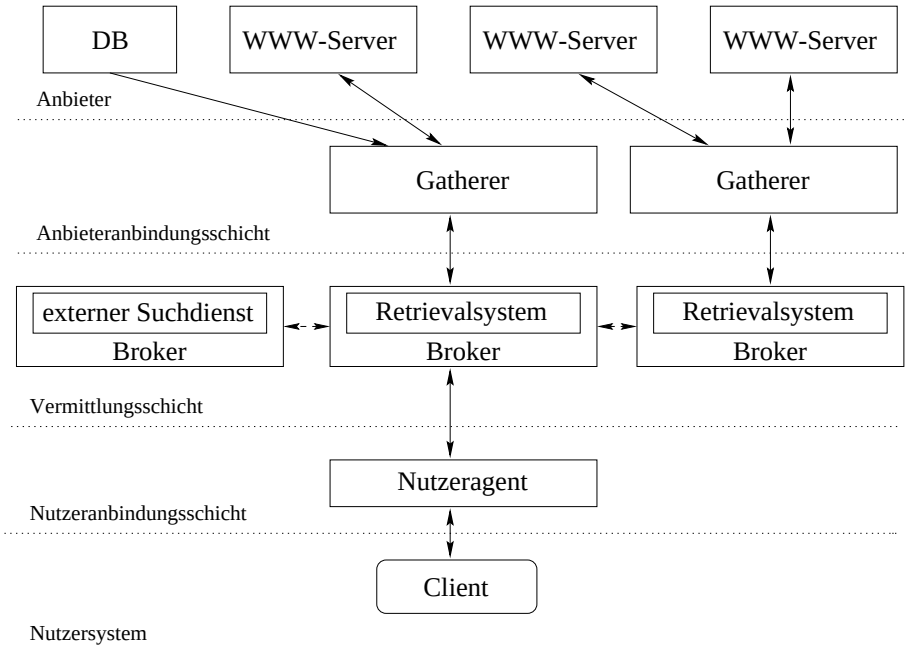

Abbildung 4.1: Architekturentwurf

Nutzersystem. Die Kommunikation der Nutzer mit dem Suchdienst erfolgt mit Hilfe des Nutzersystems, das beispielsweise ein WWW-Client sein kann.

Nutzeranbindungsschicht. Durch die Nutzeranbindungsschicht wird dem Anwender eine Schnittstelle zum Suchdienst bereitgestellt. Hier erfolgt die Transformation der Anfrage, der Aufruf der Anfrageverarbeitung und die Aufbereitung der Ergebnisse. In Abhängigkeit von der Auslastung des Nutzeragenten, der die einzige Komponente dieser Schicht darstellt, könnten auch mehrere Replikate zum Einsatz kommen. Der Nutzeragent kann gegebenenfalls eine Transparenz realisieren, die die Verwendung von mehreren Datenbasen vor dem Nutzer verbirgt.

Vermittlungsschicht. In der Vermittlungsschicht ermitteln kooperierende Broker, die jeweils über ein eigenes Retrievalsystem verfügen, die Anfrageergebnisse und leiten sie an den Nutzeragenten weiter. Die Retrievalsysteme können durch einen oder mehrere Gatherer gespeist werden. Der Nutzer stellt unter Zuhilfenahme des Nutzeragenten die Anfrage an einen Broker;

dieser ist dann für die Kommunikation mit den anderen Brokern zur Ergebnisermittlung zuständig.

Anbieteranbindungsschicht. Durch die Anbieteranbindungsschicht wird eine Datenunabhängigkeit vom zu indexierenden Dokument-Typ erreicht, die eine flexible Integration von neuen Dokument-Typen durch die Sammelkomponente, den Gatherer, gewährleistet.

Anbieter. Die Anbieter gliedern sich in Datenbanken, die über Suchformulare und die entsprechenden Gateways an das WWW angebunden sind, und WWW-Server, auf denen Nutzer ihre Dokumente in vielen Formaten bereitstellen.

In den folgenden Abschnitten werden die einzelnen Komponenten näher beschrieben.

4.3 Die Komponenten

Die Komponenten des Systems liegen verteilt vor, d.h. Gatherer, Broker und Nutzeragent können auf verschiedenen Servern laufen.

4.3.1 Der Gatherer

Das Kernstück der Informationsgewinnung bildet der Gatherer, der für die Extraktion von Daten und Metadaten aus den Dokumenten verantwortlich ist. Er realisiert die Funktionalität der Anbieteranbindungsschicht und stellt somit die Schnittstelle zwischen den Informationsanbietern und Informationsvermittlern bereit. Der Harvest-Gatherer stellt vollständig die von der zu konzipierenden Suchmaschine benötigte Funktionalität bereit, so daß in diesem Abschnitt lediglich zwei Fragen bearbeitet werden müssen:

- Wie sollen die Indizes verteilt werden ?
- Wie kann der Inhalt von Datenbanken (bzw. daraus dynamisch generierte Dokumente) der Suchmaschine zugänglich gemacht werden ?

Wie sich zeigen wird, können beide Fragen auf relativ einfache Art beantwortet werden.

4.3.1.1 Die Indexverteilung

Die Gatherer laufen lokal auf Providerseite um einerseits die Netzwerkbelastung und andererseits die Bearbeitungszeit niedrig zu halten.[3] Diese Lokalität wird in

[3]Es gibt jedoch Fälle, in denen Dokumente via HTTP oder FTP Protokoll geladen werden müssen, da auf der entsprechenden Web-Site kein Gatherer läuft.

Abbildung 4.2 durch die Kapselung mittels gestrichelter Linie angedeutet.[4] In

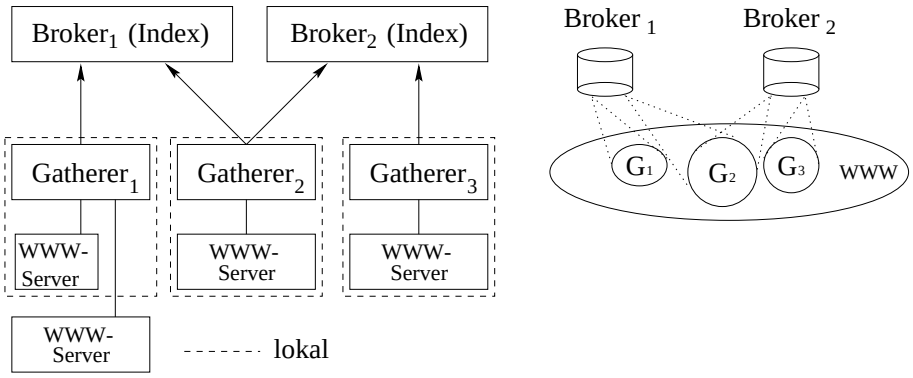

Abbildung 4.2: Gemeinsame Nutzung eines Gatherers

periodischen Abständen werden die verfügbaren Dokumente auf Veränderungen geprüft, um die Aktualität der Daten zu gewährleisten. Der Gatherer ermittelt aus den Dokumenten Informationen die den Inhalt betreffen (z.B den Titel) sowie Meta-Informationen, zu denen bspw. URL, Größe und Datum der letzten Änderung gehören können. Diese Informationen werden in regelmäßigen Abständen vom Broker angefordert, der diese in das Retrievalsystem einspeist. Um eine wiederholte Bearbeitung der lokalen Dokumente durch andere Gatherer zu vermeiden, besteht die Möglichkeit, daß mehrere Broker durch einen Gatherer gespeist werden. Wie aus Abbildung 4.2 weiterhin ersichtlich ist, nutzen die beiden Broker jeweils einen eigenen Gatherer ($Gatherer_1$ bzw. $Gatherer_3$) sowie einen gemeinsamen ($Gatherer_2$). Die daraus resultierenden Indexierungsbereiche der Broker (entspricht den Web Views aus Abschnitt 2.2) sind auf der rechten Seite der Abbildung dargestellt, wobei G_1 bis G_3 jeweils die Menge der Dokumente ist, die durch $Gatherer_1$ bis $Gatherer_3$ erfaßt sind.

Die Übertragung dieses Konzeptes auf die bereits bestehende Struktur von MV-Info bietet mehrere Ansätze. Bisher sind die Gatherer jeweils an eine oder mehrere Domänen gebunden (Abbildung 4.3), die sie indexieren.[5] Es werden demnach Digital Neighbourhoods des Typs DDNL (Digital Distance by Network Location) gebildet. Die Ergebnisse der Gathererläufe werden wöchentlich vom Broker eingesammelt und in einem Retrievalsystem gespeichert. Somit bietet sich eine Verteilung der gesammelten Indexe auf diverse Broker, die jeweils die Dokumente mehrerer Domänen verwalten, an. Demzufolge muß bei diesem Ansatz „lediglich“ eine geeignete Zusammenfassung von Domänen gefunden werden.

In Abbildung 4.4 wird ein weiterer Ansatz dargestellt. Hierbei werden alle Dokumente, die einer Thematik zugeordnet werden können, in einem Index

[4] In den weiteren Abbildungen wird auf diese Lokalität nicht weiter hingewiesen da sie für die Betrachtungen nicht von Belang sind.

[5] z.B. Gatherer uni-hro: `*.uni-rostock.de` [Sg98]

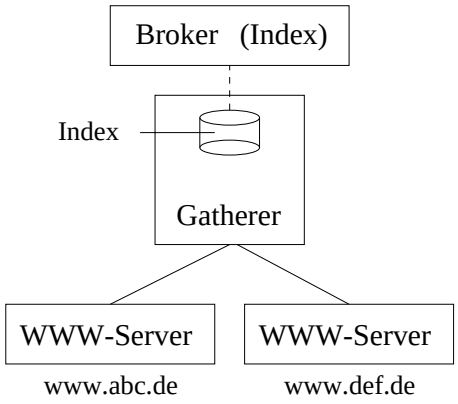

Abbildung 4.3: Zusammenfassung der Indizes nach DDNL: *Nutzung der vorhandenen Gathererstruktur*

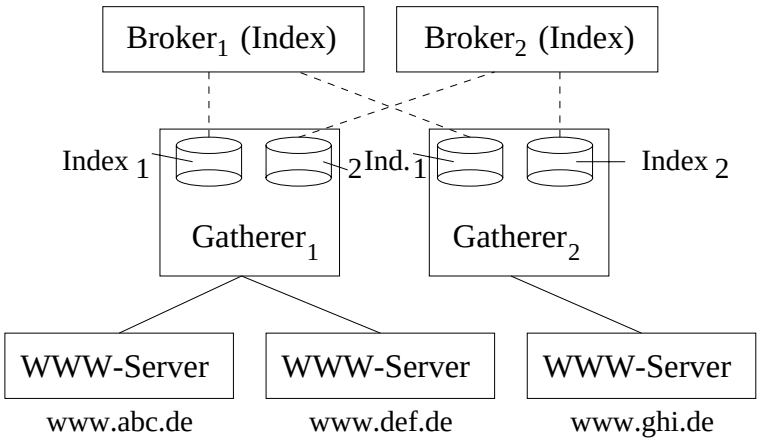

Abbildung 4.4: Zusammenfassung der Indizes nach DDC: *Die Gatherer erzeugen je Thematik einen Index (Index$_1$ und Index$_2$), der vom Broker, der die zu einer Thematik gehörenden Dokumente indiziert, gelesen wird.*

zusammengefaßt. Dies entspricht der Digital Neighbourhood des Typs DDC (Digital Distance by Content); ein oder mehrere Themengebiete werden zu einem Index zusammengefaßt und von einen Broker durchsucht. Die Realisierung dieses Ansatzes ist im Vergleich zur ersten Alternative wesentlich aufwendiger, da die Einordnung der Dokumente in die jeweilige Thematik zu erfolgen hat. Um die bestehenden Strukturen nutzen zu können, sind zusätzlich zu den bereits durchgeführten Gathererläufen weitere Operationen notwendig, d.h. im ersten Schritt sind die Domänen zu durchsuchen und im zweiten Schritt die Dokumente in die entsprechende Thematik einzuordnen. Das Ergebnis wären entweder a) mehrere lokale, themenspezifische Indizes (Abbildung 4.4) oder b) ein lokaler, themenunabhängiger Index, in dem zusätzlich für jedes Dokument die zugehörigen Thematiken gespeichert sind (Abbildung 4.5). Die Vorgehensweise b) ist eigentlich nicht

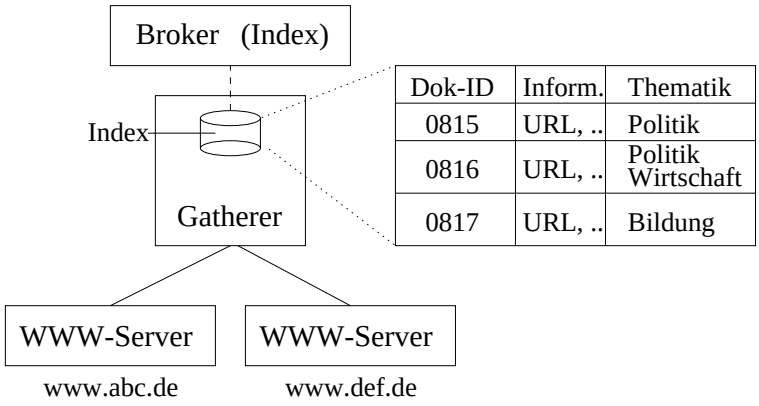

Abbildung 4.5: Zusammenfassung der Indizes nach DDC: *Der Gatherer erzeugt einen Index, in dem zu jedem Dokument zusätzlich die dazugehörigen Thematiken gespeichert sind. Eine Verteilung nach Thematiken findet nicht statt.*

im Sinne der Definition der Digital Neighbourhood des Typs DDC, da alle Dokumente einer Domäne unabhängig vom Inhalt in einem Index zusammengefaßt werden.[6] Weiterhin hätte sie Auswirkungen auf die Anfragesprache des Brokers, da bei Anfragen zusätzlich eine Selektion (innerhalb der Datenbank) nach der Thematik erforderlich ist, sofern diese bekannt ist. Beim Vorgehen nach a) findet die Selektion bereits durch die Auswahl der zu kontaktierenden Broker statt. Zusammenfassend betrachtet liefert der Ansatz der themenorientierten Indexierung allein keine großen Vorteile, denn bei der Vorgehensweise a) ist keine verteilte Anfragebearbeitung nötig (sofern sich die Suche auf ein Themengebiet beschränkt) und der Bearbeitungsaufwand durch den Gatherer höher als beim ersten Ansatz. Weiterhin ist bei der Einordnung eines Dokumentes in mehrere Thematiken eine mehrfache Speicherung erforderlich. Die Vorgehensweise bei b) hingegen kann in Verbindung mit der DN des Typs DDNL durchaus Vorteile bringen. Zu jedem Index ist dann nämlich bekannt, wieviele Dokumente er zu den einzelnen Themenbereichen enthält. Dieses Wissen kann bei der Auswahl der Broker in der Anfrageverarbeitung genutzt werden. Bei genauerer Betrachtung der themenorientierten Zusammenfassung zeigt sich aber ein unerwünschter Nebeneffekt. Um die Einordnung der Dokumente in die Thematik treffen zu können, müssen dem Gatherer und dem Broker die verfügbaren Thematiken bekannt sein. Änderungen und Erweiterungen in den Thematiken führen weitreichende Auswirkungen in den Komponenten nach sich. Bei Vorgehensweise a) müßte von jedem Gatherer ein zusätzlicher Index erzeugt werden; bei Vorgehensweise b) müßte in der Anfrage die Wertemenge des Attributes, das die Selektion der Thematik angibt, in jedem Broker geändert werden. Deshalb erscheint die Bildung von themenorientierten Digital Neighbourhoods eine zu enge Bindung der Suchmaschine an die Struktur

[6] Formal betrachtet würde der Radius R (siehe Definition 5 Abschnitt 2.2) in diesem Fall den maximalen Wert annehmen.

des MV-Info Kataloges zu erfordern.

Die in Abschnitt 2.2 ebenfalls definierte Digital Neighbourhood des Typs DDGL (DD by Geographic Location) läßt sich nicht umsetzen, da nicht davon ausgegangen werden kann, daß die geographische Lage des Servers mit der geographischen Lage des Informationsanbieters übereinstimmt. So kann ein Anbieter seine Daten auf dem Server eines Internet-Providers gespeichert haben, der sich in einer ganz anderen Stadt befindet. Die Bildung von Digital Neighbourhoods des Typs DDHTL (DD by Hypertext Location) scheint ebenfalls kein passender Ansatz zu sein.
Somit ist die Frage der Indexverteilung beantwortet und die erste Anforderung an die Suchmaschine realisiert: die **verteilte Indexierung**.

4.3.1.2 Integration von Datenbankanfragen

Eine weitere Anforderung an die zu konzipierende Suchmaschine ist die Integration von Datenbankanfragen, d.h. es sollen auch dynamische Dokumente, die aus Datenbankinhalten generiert werden, von der Suchmaschine indexiert werden.[7] Durch die offenen Schnittstellen des Gatherers ist es einfach möglich, für jeden Dokument-Typ ein Programm anzugeben, daß aus dem Dokument die gewünschten Informationen extrahiert. In Abbildung 4.6 wird dieser Prozeß grob skizziert.

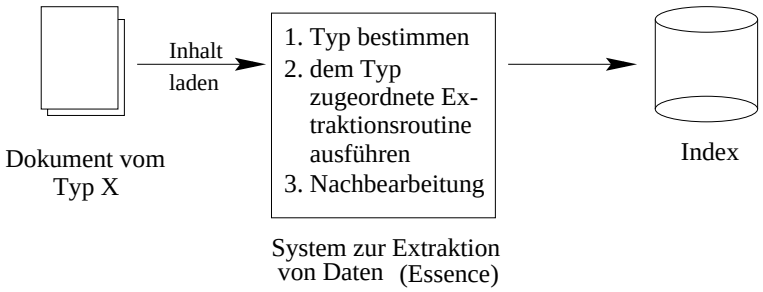

Abbildung 4.6: Der Prozeß der Datenextraktion

Die folgenden Schritte sind einmalig vor dem Sammeln der Daten auszuführen, wobei in *diesem Font* ein konkretes Beispiel[8], in dem teilweise der Inhalt einer Hoteldatenbank vom Gatherer indexiert werden soll, angegeben wird:

- Es ist ein Dokument eines neuen Typs anzulegen. *Für die Hotel-Datenbank ist dieses Dokument in Anhang B.1 angegeben.*

[7] Dies ist natürlich nicht für alle Datenbanken sinnvoll, sondern vom Inhalt abhängig.

[8] Die Realisierung erfolgte durch Herrn Dipl.-Inf. Gunnar Weber (`weber@informatik.uni-rostock.de`).

- Dieser neue Dokument-Typ ist dem System *Essence*[9], das für die Extraktion zuständig ist, in den entsprechenden Konfigurationsdateien mitzuteilen sowie eine Vorschrift anzugeben, mit deren Hilfe Dokumente dieses Typs erkannt werden.[10] *In der Konfigurationsdatei* `lib/byname.cf` *wird Essence mitgeteilt, daß der neue Dokument-Typ durch die Dateiendung* `ing-db` *erkannt werden kann.*

- Anschließend ist für diesen Dokument-Typ ein Summarizer zu schreiben, der aus Dokumenten dieses Typs „Zusammenfassungen" generiert. *Der Summarizer liest das Dokument, in dem die aus der Datenbank zu extrahierenden Attributwerte, Informationen zum Zugriff auf die Datenbank (Relationenname) und der URL des zugehörigen Suchformulares gespeichert sind, ein und generiert daraus ein SOIF-Fragment.*[11] *Im Anhang B.2 ist dieses Fragment abgebildet.*

- Wenn nötig kann ein Programm angegeben werden, das eine Nachbearbeitung der extrahierten Informationen durchführt. *Im letzten Schritt muß nun ein Mapping der URL stattfinden, um dem Suchenden als Ergebnis seiner Anfrage das Suchformular und nicht das vom Gatherer gelesene* `ing-db`*-Dokument zu präsentieren. Der daraus resultierende Dokument-Index für die Hotel-Datenbank ist in Anhang B.3 dargestellt.*

Damit ist auch die zweite Frage, wie der Inhalt einer Datenbank der Suchmaschine zugänglich gemacht werden kann, beantwortet und eine weitere Anforderung an die verteilte Suchmaschine gewährleistet: die **Integration von Datenbankanfragen**.

4.3.2 Der Broker

Wie aus der Architektur des Harvest-Systems deutlich wurde, ist der Harvest-Broker für die Anfrageverarbeitung, Transformationen und die Einbindung des Retrievalsystems verantwortlich. Für die Realisierung einer verteilten Anfragebearbeitung reicht die Funktionalität aber nicht aus. Deshalb wird der in diesem Abschnitt vorgestellte Entwurf des Brokers vollkommen unabhängig vom existierenden Harvest-Broker betrachtet. Die Nutzung des Harvest-Brokers als "Retrievalsystem" ist jedoch weiterhin möglich.
Bei dem in Abbildung 4.7 dargestellten Entwurf handelt es sich um ein Grundgerüst, das sich aus diversen Modulen zusammensetzt. Eine Replikation des Sy-

[9] Essence [HS93] ist ein Bestandteil des Harvest-Gatherers [HSW96], der den Dokument-Typ ermittelt und den zugeordneten Extraktionsalgorithmus (der Summarizer genannt wird) auf das Dokument anwendet.

[10] Eine solche Vorschrift könnte beispielsweise die Dateiendung sein.

[11] In Anhang A.1 wird die dem SOIF zugrunde liegende Grammatik beschrieben und in Anhang A.2 beispielhaft erläutert.

stems wird zur Zeit nicht betrachtet. Der Prozeß der verteilten Anfragebearbeitung durch den Broker läuft in folgenden Schritten ab:

1. Empfang einer Suchanfrage durch einen Broker (Prinzipiell kann jeder Broker durch einen Nutzeragenten befragt werden, da er die volle Funktionalität bietet.)
2. Ermittlung der vielversprechendsten Broker aufgrund von Meta-Informationen über deren Datenbestände
3. Anfrageweiterleitung an diese Broker
4. Sammlung und Mischung der Ergebnisse zu einer Liste
5. Rückgabe der Ergebnisse an den Nutzeragenten

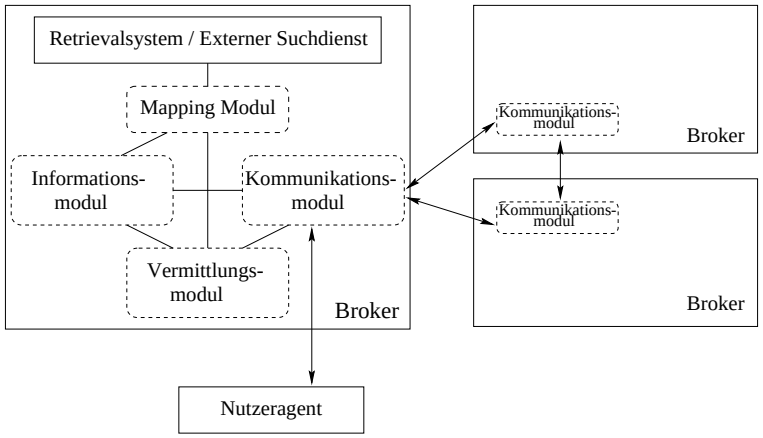

Abbildung 4.7: modularer Aufbau des Brokers:*Durch die Einführung des Kommunikationsmodules ist die Zusammenarbeit der Broker möglich geworden.*

In den kommenden Abschnitten werden die Module vorgestellt und ihre grobe Funktionalität festgelegt. Auf eine Festlegung von Schnittstellen wurde verzichtet, da der Prozeß der Softwareentwicklung nicht Bestandteil dieser Arbeit ist.

4.3.2.1 Die Module

Durch die Modularisierung des Brokers und die Schaffung von wohldefinierten Schnittstellen wird eine Offenheit des Systems erreicht, die eine leichte Erweiterbarkeit der Module gestatten soll. Die Anbindung des Brokers an das Retrievalsystem bzw. den externen Suchdienst ist Aufgabe des in Abschnitt 4.3.2.3 betrachteten **Mapping Modules**. Das in Abschnitt 4.3.2.2 vorgestellte **Informationsmodul** liefert Informationen über den Datenbestand des Brokers und das zugrunde liegende Retrievalsystem, die für die verteilte Anfragebearbeitung benötigt werden. Aufgrund dieser Daten ist das in Abschnitt 4.3.2.4 vorgestellte **Vermittlungsmodul** in der Lage, die vielversprechendsten Broker für die

Anfragebearbeitung auszuwählen. Die Kommunikation der Broker untereinander erfolgt mit Hilfe des in Abschnitt 4.3.2.5 eingeführten **Kommunikationsmodules**.

4.3.2.2 Das Informationsmodul

Damit eine verteilte Anfragebearbeitung stattfinden kann, benötigt der vom Nutzeragenten kontaktierte Broker Informationen über den Datenbestand der an die Broker angeschlossenen Retrievalsysteme und die Broker selbst.[12] Diese Informationen werden vom Informationsmodul verwaltet und bei Bedarf geupdated. In welchen Intervallen die Informationen geupdated werden müssen, hängt davon ab, wie oft ein Broker durch die zugewiesenen Gatherer gespeist wird. Welche Informationen der Broker zur globalen Anfragebearbeitung benötigt, wird vom Konzept der Brokerauswahl und der Ermittlung des globalen Rankings bestimmt (siehe Abschnitt 4.3.2.4). Die noch zu klärende Frage ist die Darstellung dieser Informationen. In dieser Arbeit werden zwei Ansätze betrachtet: das RDF und das SOIF.

Das vom World Wide Web Consortium (W3C)[13] zur Zeit entwickelte *Resource Description Framework* (RDF) dient zur Beschreibung von Daten, die im WWW verfügbar sind. Eines der möglichen Einsatzgebiete von RDF liegt in der Beschreibung von Ressourcen (die in dieser Studienarbeit bisher als Datenbasen bezeichnet wurden) und Dokumenten um die Funktionen von Suchmaschinen zu verbessern. Da eine Kurzbeschreibung des RDF in dieser Arbeit nicht gegeben werden kann (die Entwicklung ist noch nicht abgeschlossen; es existieren zu diesem Zeitpunkt - Ende September 1998 - kaum Implementierungen; der Umfang der Studienarbeit würde um ein Drittel anwachsen), sei auf die vorläufigen "working drafts" [LS98] (RDF Modell und Syntaxbeschreibung) und [BGL98] (Schemaspezifikation) verwiesen. Obwohl die Verwendung von RDF eine spätere Anbindung von anderen "externen" Ressourcen vereinfachen würde und eine Standardschnittstelle geschaffen wäre, kann RDF in diesem Realisierungsvorschlag keine Verwendung finden.

Den zweiten Ansatz bietet das bereits vorgestellte SOIF. Die Verwendung des SOIF in der Suchmaschine bietet einige Vorteile:

- es ist im Vergleich zu RDF einfach aufgebaut

[12] Wie ist der Broker erreichbar ? Kurzbeschreibung des Inhaltes, häufig vorkommende Begriffe, ...

[13] Das W3C (`http://www.w3.org`) wurde gegründet, um die Entwicklung des WWW voranzutreiben. Durch das Design von allgemeinen Protokollen soll die Entwicklung des WWW gefördert und die Interoperabilität sichergestellt werden. Das Konsortium stellt die Informationen über das WWW für Entwickler und Anwender zur Verfügung; veröffentlicht Referenzcode-Implementierungen um Standards darzustellen und voranzutreiben; entwickelt Prototypen und Beispielanwendungen um die Anwendung von neuer Technologie zu demonstrieren. Kurz: *"Leading the Web to its Full Potential ..."*. [w3c]

- der Transport der Beschreibungen kann auf einfache Art und Weise im Body einer Nachricht erfolgen (Abschnitt 4.3.2.5) während bei RDF die Beschreibung, wie RDF in HTTP-Headern transportiert werden kann, bisher nicht festgelegt ist und eine Einbettung in ein HTML-Dokument mittels XML[14] für den Austausch von Metadaten zwischen den Brokern die Implementierung des Kommunikationsmodules aufwendiger gestalten würde
- die Darstellung von Daten erfolgt homogen in allen Komponenten (Gatherer-Broker-Nutzeragent)

aber auch einige Nachteile:

- RDF ist mächtiger (es lassen sich Relationen, Mengen, geordnete Liste, Alternative uvm. modellieren)
- RDF wird einen allgemein gültigen Standard bilden

An dieser Stelle eine Entscheidung zu treffen fällt recht schwer. Aufgrund der relativ beschränkten Nutzung von externen Ressourcen beim Einsatz der Suchmaschine bei MV-Info scheint das SOIF die bessere Wahl zu sein. Dafür spricht auch, daß die Verwendung des RDF (und der diversen Darstellungsmöglichkeiten) einen weitaus größeren Aufwand bei der Implementierung der einzelnen Komponenten erfordert und es sich bei RDF noch um eine "working draft" handelt.

4.3.2.3 Das Mapping Modul

Das Mapping Modul stellt die Schnittstelle zwischen Broker und zugrunde liegendem Retrievalsystem bzw. Suchdienst dar. Es ist für die Konvertierung der Anfragen in die Ziel-Anfragesprache (siehe auch Abschnitt 4.3.3), die Konvertierung der Anfrageergebnisse in das global verwendete SOIF (die Kommunikation der Module untereinander erfolgt durch Nachrichten in diesem Format) und die Konvertierung aller anderen notwendigen Nachrichten an das Retrievalsystem zuständig. Durch die Schaffung dieser Schnittstelle ist die Integration von beliebigen Retrievalsystemen möglich, die idealerweise drei Anforderungen erfüllen sollten:

- Ranking der Anfrageergebnisse (Vektorraummodell)
- statistischer Überblick (Worthäufigkeiten, ...) über den Inhalt der indexierten Dokumente um ein globales Ranking zu ermöglichen
- einfache Integration der von den Gatherern extrahierten Daten, die im SOIF vorliegen

[14]XML: Extensible Markup Language; beschreibt eine Klasse von Datenobjekten (XML-Dokumente) und teilweise das Verhalten von Programmen, die diese XML-Dokumente verarbeiten. Die Definition der XML erfolgt in [xml].

Durch diese Anforderungen soll der Integrationsaufwand der Retrievalsysteme in den Broker möglichst gering gehalten werden. Der Harvest-Broker bietet dafür eine sehr gute Voraussetzung, da er bereits zwei Anforderungen erfüllt und keine umfangreichen Modifikationen am System notwendig sind. Inwieweit „richtige" Retrievalsysteme diese Anforderungen erfüllen, müßte genauer untersucht werden. Aus Zeitgründen kann im Rahmen dieser Studienarbeit eine solche Untersuchung nicht stattfinden.

Aber auch die Integration von existierenden Suchdiensten ist durch das Modul möglich. In diesem Fall entfällt die Anforderung an die einfache Integration der extrahierten Daten, da diese Suchdienste über eine eigene Datengewinnung verfügen. Als problematisch kann sich jedoch die Informationsgewinnung über die in diesen Diensten indexierten Dokumente herausstellen, da viele die benötigten Informationen nicht bereitstellen.

4.3.2.4 Das Vermittlungsmodul

Das Kernstück des Brokers bildet das Vermittlungsmodul. Es ist für die lokale Bearbeitung der Anfrage (Weiterleitung an das Mapping Modul), die Rückgabe der Ergebnisse (Aufruf des des Kommunikationsmoduls) an den Homebroker[15], die Bestimmung der zu kontaktierenden Broker (Nutzung des Informationsmoduls) und das globale Ranking (nur vom Homebroker durchzuführen) zuständig. Die Ermittlung der bei einer Anfrage zu kontaktierenden Broker bildet neben dem globalen Ranking das komplexeste Problem der gesamten hier vorgestellten Konzeption. Ein konkreter Realisierungsvorschlag soll an dieser Stelle nicht gegeben werden, da die Entscheidung für eines der Verfahren tiefere Einarbeitung in die jeweiligen Konzepte erfordert.

Brokerauswahl Als erstes sollen an dieser Stelle einige Konzepte für die Ermittlung der zu kontaktierenden Broker vorgestellt werden. Verallgemeinert betrachtet, handelt es sich um ein "Text Database Discovery Problem"[16] [GGMT94], da bei der hier konzipierten Suchmaschine jeder Broker mit einer Datenbank verknüpft ist. Die Auswahl der Broker erfolgt demzufolge aufgrund von Meta-Informationen über die dem Broker zugeordnete Datenbank. Die Konzepte lassen sich grob in folgende Klassen einteilen:

- der Nutzer wählt die zu kontaktierenden Datenbanken aus (trivialer Fall)
- die Ermittlung der zu kontaktierenden Datenbanken erfolgt aufgrund zentral gehaltener Metadaten (über die Datenbanken) in einer "Master Datenbank"

[15] Der Homebroker ist der vom Nutzeragenten direkt aufgerufene Broker.

[16] Die Lösung dieses und des allgemeineren "Database Discovery Problems" ist Bestandteil aktueller Forschungen. Beim "Text Database Discovery Problem" wird angenommen, daß es sich bei allen Datenbanken um Text-Datenbanken handelt.

- die Ermittlung erfolgt ausgehend vom "Homebroker" (dem vom Nutzeragent kontaktierten Broker) dezentral

Stellvertretend für jede Klasse werden nun einige Ansätze beschrieben:

- **Auswahl erfolgt durch den Nutzer.** Eine triviale Lösung bildet die Auswahl der zu kontaktierenden Retrievalsysteme durch den Nutzer. Diesen Ansatz wählen beispielsweise einige Meta-Suchmaschinen. Der Nutzer benötigt in diesem Fall Wissen über die Systeme, die kontaktiert werden können. Dieses Wissen umfaßt den Umfang des Datenbestandes (mehr Dokumente erfaßt → mehr Treffer), die Regionalität (Datenbestand beschränkt sich auf deutsche Seiten ? → deutsche Suchbegriffe erfolgreicher), das Antwortzeit-Verhalten uvm. Da dieses Wissen nicht immer gegeben ist, werden häufig alle Systeme befragt.
- **Auswahl erfolgt zentral.**
  - Die Auswahl der in Abschnitt 3.4 vorgestellten freeWAIS-Systeme basiert auf dem "directory of servers". Hierbei handelt es sich um eine spezielle Datenbank, in der zu den frei zugänglichen Datenbanken eine kurze Zusammenfassung des Inhaltes und die 50 am häufigsten auftretenden Wörter gespeichert sind.[17] Anhand dieser Beschreibungen werden die Datenbanken bewertet und die geordnete Liste dem Nutzer für die weitere Anfragebearbeitung übermittelt.
  - Einen anderen Ansatz wählt der im Rahmen des "Stanford Digital Library projects"[18] entwickelte Dienst GlOSS (Glossary-of-Servers Server), der ausgehend von einer Anfrage die angeschlossenen Datenbanken mit Hilfe von Schätzungen bewertet und dem Nutzer eine (nach Anzahl der zu erwartenden Dokumente in der Datenbank) geordnete Liste zurückliefert. Die Schätzungen beruhen auf dem Vokabular (Begriffe, die in den Dokumenten auftauchen), der Anzahl der Dokumente, in denen der Begriff auftaucht, und der Anzahl der Dokumente in der Datenbank. Um diese Schätzungen durchführen zu können, extrahiert GlOSS diese Werte in periodischen Abständen aus den Datenbanken und speichert sie in einer zentralen Datenbank. Wie die Schätzungen für Datenbanken, deren Ergebnisermittlung auf dem Booleschen Retrieval basiert (siehe Abschnitt 2.3.1), erfolgen, wird in [GGMT94] beschrieben. Schätzungen für Datenbanken, deren Ergebnisermittlung auf dem Vektorraummodell (siehe Abschnitt 2.3.2) basieren, erfolgen

[17] Beim Indexieren der Dokumente muß explizit angegeben werden, daß die Datenbank im "directory of servers" registriert werden soll.

[18] Projektaufgabe der Universität Stanford war die die Schaffung einer Infrastruktur, die Interoperabilität zwischen heterogenen, autonomen elektronischen Bibliotheksdiensten gewährleistet. (`http://www-diglib.stanford.edu/diglib/pub/`)

mit Hilfe von gGlOSS (generalized Glossary-of-Servers Server), einer Weiterentwicklung von GlOSS. Eine Beschreibung des dort verwendeten Verfahrens ist in [GGM95] zu finden.

- **Auswahl erfolgt dezentral.** Im Entwurf des MEDOC-IVS wurde ein dezentraler Ansatz für die Selektion der Broker vorgestellt. Für die Annahme, daß zu festgelegten Kosten eine maximale Anzahl von Dokumenten ("Anzahlmaximierung") gefunden werden soll, sind in [BDG+96] einige Betrachtungen angestellt worden. Ein theoretischer Ansatz zur Auswahl von Brokern ist in [Fuh98] beschrieben.

Bei Verwendung eines zentralen Ansatzes muß in den Entwurf der Vermittlungsschicht entweder eine weitere Komponente eingefügt werden, die für die zentrale Verwaltung der Broker-Metadaten zuständig ist, oder jeder Broker muß diese Daten lokal - im Informationsmodul - halten. Letztere Vorgehensweise stellt höhere Anforderungen an die Ressourcen, da bei n Brokern jeder Broker n^2 Metadaten über die Broker verwalten muß. (Der Aufwand für die Wartung der Metadaten ist dementsprechend hoch, da die regelmäßigen Updates allen Brokern bekannt gemacht werden müssen.) Für den dezentralen Ansatz spricht die Unabhängigkeit von einer zentralen Instanz, die einen Flaschenhals bilden kann und im schlechtesten Fall das ganze System zum Erliegen bringt.

Bei der Nutzung der hier konzipierten Suchmaschine im MV-Info Dienst könnte die Selektion der zu kontaktierenden Broker durch eine weitere Information vereinfacht werden. Da bekannt ist, daß Digital Neighbourhoods vom Typ DD-NL (wie in Abschnitt 4.3.1.1 vorgeschlagen) gebildet werden, kann der Nutzer die Suche auf bestimmte Domänen einschränken. Dies ist aber nur sinnvoll, wenn eine "lokal" beschränkte Suche vorgenommen werden soll. Dazu muß jeder Nutzeragent und jeder Broker Informationen über die Zuordnung von Domänen zu Brokern verwalten.[19] Eine Abwägung, ob Aufwand und Nutzen in einem vernünftigen Verhältnis stehen, ist deshalb vorzunehmen.

Globales Ranking Die zweite Hauptaufgabe des Homebrokers besteht in der Bestimmung des globalen Rankings. Jeder kontaktierte Broker, dessen Retrievalsystem die Dokumente entsprechend ihrer Ähnlichkeit zur Anfrage rankt (Vektorraummodell), liefert seine Ergebnisse an den Homebroker zurück, der sie mischt und einheitlich bewertet. Die Problematik besteht darin, daß

- die Berechnung der Ähnlichkeit auf verschiedenen Algorithmen beruhen kann

[19] Beispiel: Broker A hat Dokumente der Domänen www.abc.de und www.def.de, Broker B Dokumente der Domänen www.ghi.de und www.jkl.de indexiert. Der Nutzeragent teilt dem Nutzer mit, welche Domänen indexiert sind; dieser schränkt daraufhin seine Suche auf die Domäne www.abc.de ein $\longrightarrow$ nur Broker A muß kontaktiert werden.

- diese Algorithmen nicht bekannt sein müssen
- gleiche Rankingalgorithmen in Abhängigkeit vom Inhalt des Retrievalsystems (unterschiedliche Gewichte für den gleichen Begriff) unterschiedliche Ergebnisse liefern können

Der wohl einfachste Ansatz besteht darin, in einem Round-Robin-Verfahren aus jeder Teilergebnisliste das jeweils erste Dokument in die Ergebnisliste einzuordnen, anschließend das jeweils zweite Dokument, usw. Dieses Vorgehen garantiert leider kein besonders gutes Mischen, da das Ranking der Teilergebnisse untereinander keine Rolle spielt.[20] Da eine Forderung an das Ranking ist, daß "gute" Dokumente in der Ergebnisliste vor "weniger guten" auftauchen, kann dieses Verfahren nicht verwendet werden.

Ein anderer Ansatz besteht darin, die Dokumente der Teilergebnislisten vom Homebroker laden und einheitlich bewerten zu lassen. Da bei diesem Ansatz auf alle Ergebnisdokumente über das Netz zugegriffen werden muß, ist dies ein sehr zeitaufwendiges Verfahren. Bei hoher Netzbelastung kann dieses Verfahren demzufolge nicht angewendet werden.

Die wohl beste Lösung bietet der folgende Ansatz, bei dem weitere Informationen über das verwendete Ranking und die darin einfließenden Parameter dem Homebroker bekannt gemacht werden. Da bei MV-Info davon ausgegangen werden kann, daß diese Informationen frei verfügbar sind, sollte dieser Ansatz gewählt werden. Daß die bisher verwendeten Retrievalsysteme alle benötigten Informationen ermitteln können, scheint jedoch fraglich zu sein.[21] In [GCGMP97] sind diejenigen Metadaten aufgelistet, die ein Broker zusätzlich für jedes Ergebnisdokument zurückliefern sollte: Score (Ranking), Source (Broker-ID), Term-frequency (Anzahl der Vorkommen des Anfrageterms im Dokument), Term-weight (Gewichtung des Terms im Dokument), Document-frequency (Anzahl der Dokumente, die diesen Term enthalten), Document-size (Größe) und Document-count (Anzahl der Token im Dokument).[22] Diese Informationen können in Verbindung mit den Informationen über das Retrievalsystem[23] (Stopwortliste,...) zur Berechnung eines globalen Rankings verwendet werden. Dabei sind jedoch weitere Betrachtungen notwendig: Was passiert wenn ein Retrievalsystem nicht alle Informationen liefern kann ? Welcher Algorithmus wird zum globalen Ranking verwendet ?

[20] z.B. Das in Teilergebnisliste eins am schlechtesten bewertete Dokument kann "besser" sein als das am besten bewertete Dokument aus Teilergebnisliste zwei.

[21] Zur Zeit wird als Retrievalsystem Glimpse eingesetzt [HSW96]. Glimpse bietet kein Ranking an; das in Swing verwendete Ranking wurde auf Glimpse aufgesetzt. Ein Export der Informationen über Worthäufigkeit usw. findet nicht statt. Diese sind gegebenenfalls in einem getrennten Prozeß zu ermitteln. Der Wechsel auf ein "richtiges" Retrievalsystem, daß auf dem Vektorraummodell basiert, sollte deshalb in Betracht gezogen werden.

[22] In [GCGMP97, Abschnitt 4.2] werden die benötigten Metadaten näher beschrieben.

[23] In [GCGMP97, Abschnitt 4.3] werden diese Metadaten näher beschrieben.

4.3.2.5 Das Kommunikationsmodul

Die Kommunikation der Broker untereinander bzw. zwischen Nutzeragent und Broker wird durch das Kommunikationsmodul realisiert. Dabei werden zwischen den Komponenten zwei Arten von Nachrichten ausgetauscht:

- Nachrichten, die Anfragen bzw. Anfrageergebnisse enthalten
- Nachrichten, die Informationen über die Broker (und die daran angeschlossenen Retrievalsysteme), also Metadaten, enthalten

Dieser Ansatz wird beispielsweise bei STARTS[24] verwendet. Auch die dort gewählte Beschreibung der Anfragen, Anfrageergebnisse und Source-Metadaten durch das SOIF bietet sich an dieser Stelle an, da so eine homogene Beschreibung der Daten im gesamten System (Gatherer-Broker-Nutzeragent) erfolgen kann. Konkrete Beispiele, wie eine Anfrage, Anfrageergebnisse oder Metadaten bei STARTS mit Hilfe des SOIF repräsentiert werden, sind in [GCGM+97] aufgeführt.

Als zugrundeliegende Transportschicht kann das HTTP Verwendung finden, da es ein sehr einfaches und flexibles Protokoll ist. Da es sich um ein synchrones Protokoll[25] handelt, besteht bereits eine Art Mechanismus, durch den eine fehlende Verfügbarkeit von Komponenten erkannt werden kann. In der prototypischen Implementierung des MEDOC-IVS wurde ebenfalls das HTTP als Transportschicht genutzt [DLM98].[26]

Ein kurzes Beispiel[27] illustriert die einfache Verwendbarkeit von HTTP als Transportprotokoll und dem SOIF zur Beschreibung der Anfrage:

```
POST / HTTP/1.0
Date: Fri, 25 Sep 1998 14:19:22 GMT
Content-type: application/x-broker-message; type="query"
Content-length: 210

@Query{
Version{5}: 0.815
QueryIdentifier{11}: b4a2c540998
```

[24] STARTS steht für "**Sta**nford Protocol Proposal for Internet **Re**trieval and **S**earch" und wurde an der Universität Stanford (Kalifornien, USA) in Zusammenarbeit mit verschiedenen Suchmaschinenherstellern entwickelt [Gra]. Ziel des Projektes war es ein Protokoll zu entwerfen, daß die Suche und das Retrieval von Informationen in verteilten und heterogenen Datenbasen gestattet.

[25] Die Verbindung bleibt solange bestehen, bis der Server geantwortet hat bzw. bis ein timeout erfolgt.

[26] Das darauf aufsetzende Kommunikationsprotokoll hingegen arbeitet asynchron. Je eine HTTP-Verbindung ist für die Übermittlung der Nachricht und die Antwort der Empfängerkomponente notwendig, um diese Asynchronität zu erreichen.

[27] Der Wert des Headers `Content-type` und die Attribute des SOIF-Objektes für die Nachricht sind hier willkürlich gewählt.

```
QueryExpression{20}: (keywords = "Hotel")
}
```

Im Header[28] dieser Nachricht wird dem Empfänger mitgeteilt, daß es sich bei dieser Nachricht um eine Anfrage handelt (`x-broker-message; type="query"`). Im Body dieser Nachricht befindet sich ein SOIF-Objekt vom Typ `Query`, dessen Attribute die Version des Systems (`0.815`), die Anfrage-ID (`b4a2c540998`) und die Anfrage (`keywords = "Hotel"`) beschreiben.

Zusammenfassend läßt sich sagen, daß durch das Kommunikationsmodul einerseits einen WWW-Client zum Absenden der Nachrichten und andererseits ein WWW-Server zum Empfangen der Nachrichten realisiert werden muß. Da es für einige Programmiersprachen bereits Bibliotheken gibt (Python [pyt], Perl [per], C, ...), die die Funktionalität eines WWW-Clients und eines WWW-Servers implementieren, ist dieser Ansatz relativ einfach umzusetzen.

4.3.3 Der Nutzeragent

Der Nutzeragent dient als Schnittstelle zwischen Suchdienst und Anwender (Client). Er ist für die Transformation der Ein- und Ausgaben, den Aufruf eines Brokers und die Zwischenspeicherung der Anfrageergebnisse zuständig. Jedem Nutzeragenten ist ein Broker fest zugeordnet; sollte dieser ausfallen, kann anhand einer Liste[29] vom Nutzeragenten ein alternativer Broker ausgewählt und kontaktiert werden. Dadurch wird die Verfügbarkeit des Suchdienstes im Fall, daß ein Broker nicht erreichbar ist, signifikant erhöht. Die Zwischenspeicherung von Anfrageergebnissen auf Seiten des Nutzeragenten führt zu einer Entlastung des Brokers, da bei umfangreichen Ergebnislisten die seitenweise Aufbereitung für die Darstellung (Dokument 1..20, 21..40, usw.) nicht durch den Broker erfolgen muß.

Die zur Zeit bestehende Struktur des Nutzeragenten (Abbildung 4.8) bei Swing unterteilt sich in zwei Komponenten: den eigentlichen Nutzeragenten, der dem Nutzer die Suche gestattet und den Abonnementdienst, der dem Nutzer weitere Funktionen bereitstellt. Dieser Abonnementdienst wurde im Rahmen einer Studienarbeit konzipiert und prototypisch implementiert [Por98]. Er stellt dem Nutzer folgende Funktionen bereit:

- Benachrichtigung über Veränderungen im System
- zyklisch wiederkehrende Anfragebearbeitung
- turnusmäßige Bereitstellung von Informationen zu einem speziellen Thema

[28] Eine Beschreibung der HTTP-Header ist in [Klu96, Kapitel 5] zu finden.

[29] Diese Liste kann manuell oder automatisch gepflegt werden. In letzterem Fall ist entweder eine Kommunikation der Nutzeragenten untereinander nötig, oder eine zentrale Komponente für die Verwaltung der Liste aller Broker zuständig.

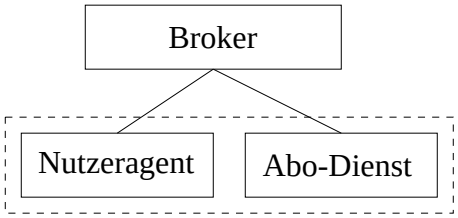

Abbildung 4.8: Struktur des Nutzeragenten bei Swing: zur Zeit zwei getrennte Komponenten; eine spätere Verschmelzung beider Komponenten zu einer (gestrichelte Linie) ist denkbar

Somit kann für jeden Nutzer ein individuelles Profil erstellt werden, das seinem Informationsbedürfnis entspricht.
Um die bestehenden Komponenten für die eigene Realisierung nutzen zu können, sind einige Modifikationen (Austauschformat der Daten) notwendig. Die oben geforderte Funktionalität (Transformation, Aufruf des Brokers, Zwischenspeicherung der Anfrageergebnisse) hingegen wird bereits vollkommen erfüllt.

Anfragesprache Eine letzte zu klärende Frage betrifft die Anfragesprache. Dem Nutzer wird ein Formular präsentiert, in dem die Suchbegriffe eingetragen werden. (Die Eingabe einer Anfrage in der Anfragesprache ist nicht empfehlenswert, da die Zielgruppe überwiegend aus "erfahrenen Laien" besteht und eine Anpassung an bestehende "Standards" gewährleistet sein sollte.) Die zugrundeliegende Anfragesprache sollte zumindest

1. die Operatoren AND, OR und NOT (z.B. distributed AND retrieval) bereitstellen und
2. attributierte Anfragen (Attribut = Wert, z.B. title=database)

erlauben. Eine Einführung von weiteren Operatoren (phonetische Suche usw.) ist ebenfalls denkbar, wobei einige Probleme auftreten können, da nicht jedes Retrievalsystem diese Operatoren unterstützen muß und eine Anwendung auf jedes Attribut nicht unbedingt sinnvoll ist (phonetische Suche nach letztem Änderungsdatum ?). Weiterhin ist zu beachten, daß nicht alle Retrievalsysteme über die gleiche Anfragesprache verfügen; eine Abbildung der hier vorgestellten Anfragesprache in die Zielsprache des Retrievalsystems ist deshalb vorzunehmen. Die dabei zu berücksichtigenden Aspekte umfassen die jeweilige Syntax (z.B. "distributed AND retrieval" vs. "+distributed +retrieval"), die unterschiedliche Benennung von Attributen (Title in Retrievalsystem A, Titel in Retrievalsystem B), die fehlende Unterstützung von strukturierten Anfragen (Lösung: Volltext-Suche), die unterschiedliche Granularität der Attribute (Vorname, Nachnache vs. Name), den Einfluß der Problemlösungen auf das globale Ranking und vieles mehr. Diese sehr komplexe Aufgabe der Abbildung ist durch das Mapping Modul zu erledigen.

4.4 Fazit

In den vergangenen Abschnitten wurde das Grundgerüst für eine verteilte Suchmaschine geschaffen und Realisierungsvorschläge für die einzelnen Komponenten gegeben. Der Gatherer und der Nutzeragent sind in der (bei Swing) bestehenden Form komplett nutzbar; beim Nutzeragenten ist jedoch eine Umstellung auf das neue Austauschformat (SOIF) notwendig. Die neu entworfenen Module des Brokers erfordern weitere Betrachtungen um die angesprochenen Probleme lösen zu können.

Kapitel 5

Abschließende Bemerkungen

5.1 Zusammenfassung

In dieser Studienarbeit wurde eine verteilte Suchmaschine konzipiert, die folgenden Anforderungen genügt:

- Datenbankinhalte können indexiert werden
- externe Suchdienste können aufgerufen werden
- verteiltes Retrieval findet statt
- die Ergebnisse werden global bewertet

Eine erste allgemeine Einführung in die Problematik des Suchens führte zur Motivation dieser Arbeit. Die anschließende Erklärung der grundlegenden Konzepte schuf die Grundlage für die Analyse der Systeme Harvest, freeWAIS, des MEDOC-IVS und der Spezifikation von Z39.50 bezüglich der oben genannten Kriterien. Ergebnis dieser Analyse war eine Auswahl der am besten zu verwendenden Konzepte. Als Basis für die Suchmaschine wurde das Harvest-System ausgewählt, da es aufgrund seiner guten Erweiterungsmöglichkeiten die besten Voraussetzungen bot.

Im Anschluß daran wurde das Grundgerüst für die Suchmaschine dargelegt und die einzelnen Komponenten vorgestellt und ihre Funktionalität beschrieben. Für den Gatherer und den Nutzeragenten konnten konkrete Realisierungsvorschläge gegeben werden, da die bereits bestehenden Softwarelösungen den Anforderungen vollauf genügen. Die Konzeption des Brokers führte zu einigen offenen Fragen, die in dieser Arbeit nicht gelöst werden konnten, da der Umfang der Arbeit sich vervielfacht hätte.

5.2 Ausblick

Um die volle Funktionalität der konzipierten Suchmaschine zu realisieren, sind weitere vertiefende Betrachtungen zu den folgenden Fragen notwendig:

Mapping Modul. Es sind geeignete Ansätze für die Abbildung der globalen Anfragesprache auf die vom Retrievalsystem verwendete Anfragesprache zu finden.

Vermittlungsmodul/verteilte Bearbeitung. Für die Ermittlung der zu kontaktierenden Broker wurde der dezentrale Ansatz vorgeschlagen. Da aber kein konkreter Lösungsvorschlag für die Umsetzung gegeben werden konnte, muß für einen ersten Prototypen gegebenenfalls auf den dezentralen Ansatz zurückgegriffen werden. Dabei ist die Entscheidung zu treffen, ob Informationen über die Indexverteilung (DN vom Typ DDNL) nutzbar sind.

Vermittlungsmodul/Ranking. Es ist ein geeignetes Rankingverfahren für die globale Bewertung zu ermitteln, in dem neben dem lokalen Rankingalgorithmus auch Probleme in der Anfragetransformation (Operator nicht im Retrievalsystem implementiert - die Anfrage wird leicht modifiziert abgearbeitet) berücksichtigt werden. Die Einführung eines "relevance feedback" kann ebenfalls in Betracht gezogen werden.

Kommunikationsmodul. Der Aufbau der Nachrichten ist festzulegen und Fehlerfälle in der Kommunikation (z.B. Broker nicht erreichbar) zu berücksichtigen.

Replikation. Eine Replikation der Broker bzw. der zugrundeliegenden Retrievalsysteme wurde nicht betrachtet, da Untersuchungen bezüglich der Auslastung der Systeme notwendig sind. Durch Replikation wird der Prozeß der Brokerermittlung aufwendiger, da nicht nur die Auswahl des Brokers sondern auch die Auswahl des Replikates erfolgen muß.

Anfragesprache. Die kurz eingeführte Anfragesprache bietet nur die elementarsten Operatoren. Für komplexere Anfragen kann deshalb eine Erweiterung um weitere Operatoren notwendig werden.

Literaturverzeichnis

[BD+95] Bowman, C. M.; Danzig, P. B.; ; R.Hardy, D.; Manber, U.; Schwartz., M. F.: The Harvest Information Discovery and Access System. *Computer Networks and ISDN Systems*, Band 28, S. 119–125, 1995.
`ftp://ftp.cs.colorado.edu/pub/cs/techreports/schwartz/Harvest.Conf.ps.Z`.

[BDG+96] Boles, D.; Dreger, M.; Großjohann, K.; lohrum, S.; Menke, D.: Architektur und Funktionalität des MEDOC-Dienstes, 1 1996.
`http://www.inf.fu-berlin.de/~medoc3/together/ivsges.ps.gz`.

[BDMS94] Bowman, C. M.; Danzig, P. B.; Manber, U.; Schwartz., M. F.: Scalable Internet Resource Discovery: Research Problems and Approaches. *Communications of the ACM*, Band 37, Nr. 8, S. 98–107, 8 1994.
`ftp://ftp.cs.colorado.edu/pub/cs/techreports/schwartz/RD.ResearchProblems.Jour.ps.Z`.

[BGL98] Brickley, D.; Guha, R.; Layman, A.: Resource Description Framework (rdf) Schema Specification, 1998.
`http://www.w3.org/TR/WD-rdf-schema/`.

[BGM96] Büll, A.; Großjohann, K.; Menke, D.: Bewertung exisierender Brokersysteme. *TUGBoat*, Band 14, Nr. 4, S. 395–422, 1996.

[DLM98] Dreger, M.; Lohrum, S.; Müller, P.: The MeDoc Communication Protocol. In: *Digital Libraries in Computer Science: The MeDoc Approach*, Lecture Notes in Computer Science 1392, S. 89–101, 1998.

[DLSZ98] Dreger, M.; Lohrum, S.; Schweppe, H.; Ziegler, C.: Ariadne, an Interactive Navigation and Search System for Computer Science Information on the World-Wide Web. In: *Digital Libraries in Computer Science: The MeDoc Approach*, Lecture Notes in Computer Science 1392, S. 127–143, 1998.

[Dre97] Dreger, M.: Replikation in der Vermittlungsschicht des M$_{\text{E}}$DOC-Systems, 1997.
http://www.inf.fu-berlin.de/~medoc3/papers/replikation.ps.gz.

[Fer98] Ferber, R.: Data Mining and Information Retrieval, 1998.
Skript zur Vorlesung an der TU Darmstadt
http://www-cui.darmstadt.gmd.de/~ferber/dm-ir/.

[Ftpa] FTP Search - other similar services.
http://ftpsearch.ntnu.no/search-info/others.html.

[Ftpb] Yahoo! Computers and Internet:Internet:FTPSites:Searching.
http://www.yahoo.com/Computers_and_Internet/Internet/FTP_Sites/Searching/.

[Fuh98] Fuhr, N.: A Decision-Theoretic Approach to Database Selection in Networked IR, 1998.
http://ls6-www.informatik.uni-dortmund.de/ir/reports/98/Fuhr-98b.html.

[Gar98] Garlipp, K.: Konzeption und Implementierung einer verteilten Suchmaschine für Technologie-Informationen im WWW, 1998. Diplomarbeit ist noch in Bearbeitung
http://wwwdb.informatik.uni-rostock.de/Lehre/Arbeiten/da-garlipp.html.

[GCGM+97] Gravano, L.; Chang, K.; Garcia-Molina, H.; Lagoze, C.; Paepcke, A.: STARTS: Stanford Protocol Proposal for Internet Retrieval and Search, 1997.
http://www-db.stanford.edu/~gravano/starts.ps.

[GCGMP97] Gravano, L.; Chang, K.; Garcia-Molina, H.; Paepcke, A.: STARTS: Stanford Proposal for Internet Meta-Searching, 1997.
http://www-db.stanford.edu/pub/gravano/1996/sigmod97.ps.

[GGM95] Gravano, L.; Garcia-Molina, H.: Generalizing GlOSS to Vector-Space Databases and Broker Hierarchies, 1995.
ftp://db.stanford.edu/pub/gravano/1995/vldb95.ps.

[GGMT94] Gravano, L.; Garcia-Molina, H.; Tomasic, A.: The Effectiveness of GlOSS for the Text Database Discovery Problem. In: *Proceedings of the 1994 ACM SIGMOD Conference*, May 1994.

[glo] BMBF-Projekt Global Info.
`http://www.global-info.org/`.

[Gö98] Gövert, N.: SFgate, 1998.
`http://amaunet.cs.uni-dortmund.de/projects/ir/SFgate/`.

[Gra] Gravano, L.: STARTS.
`http://www-db.stanford.edu/~gravano/starts_home.html`.

[GSM97] Goncalves, P. F.; Salgado, A. C.; Meira, S. L.: Digital neighbourhoods: Partitioning the web for information indexing and searching. In: *Advanced Information Systems Engineering*, Lecture Notes in Computer Science 1250, S. 289–302, 1997.

[HS93] Hardy, D.; Schwartz, M.: Essence: A resource discovery system based on semantic file indexing. In: *Proceedings of the USENIX Winter Conference*, S. 361–374, 1 1993.
`ftp://ftp.cs.colorado.edu/pub/cs/techreports/schwartz/Essence.Conf.ps.Z`.

[HSW96] Hardy, D. R.; Schwartz, M. F.; Wessels, D.: *Harvest Users's Manual*. University of Colorado at Boulder, Januar 1996. Technical Report CU-CS-743-94.

[InR95] Information Retrieval (z39.50): Application Service Definition and Protocol Specification, 1995.
`http://lcweb.loc.gov/z3950/agency/1995doce.html`.

[int98] InterDoc - Interdisziplinäre Dokumentenverarbeitung auf Basis des MeDoc-Dienstes, 1998.
`http://www-is.offis.uni-oldenburg.de/~haber/InterDoc/interdoc.html`.

[Klu96] Klute, R.: *Das World Wide Web*. Addison-Wesley Publishing Company, 1996.

[Koc96] Koch, T.: Suchmaschinen im Internet, 1996.
`http://www.ub2.lu.se/tk/demos/DO9603-manus.html`.

[Kos] Koster, M.: World Wide Web Robots, Wanderers, and Spiders.
`http://info.webcrawler.com/mak/projects/robots/robots.html`.

[LG98] Lawrence, S.; Giles, C. L.: Searching the World Wide Web. *Science*, Band 280, Nr. 5360, S. 98, 1998.
`http://www.neci.nj.nec.com/homepages/lawrence/websize.html`.

[LS98] Lassila, O.; Swick, R.: Resource Description Framework (RDF) Model and Syntax Specification, 1998. http://www.w3.org/TR/WD-rdf-syntax/.

[MA98] Meyer, J.; Appelrath, H.-J.: Design and Implementation of the MeDoc Fulltext System. In: *Digital Libraries in Computer Science: The MeDoc Approach*, Lecture Notes in Computer Science 1392, S. 21–33, 1998.

[md5] RFC 1321 - The MD5 Message-Digest Algorithm. http://search2.uni-rostock.de/Doc/RFC/rfc1321.txt.

[NSu] The Netcraft Web Server Survey. http://www.netcraft.com/survey/.

[per] Perl Website. http://www.perl.com/ Perl-Module: ftp://ftp.uni-erlangen.de/pub/source/CPAN/CPAN.html.

[Pfe95a] Pfeifer, U.: freewais-sf - building databases, 1995. http://amaunet.cs.uni-dortmund.de/projects/ir/freeWAIS-sf/fwsf_4.html.

[Pfe95b] Pfeifer, U.: WAIS: Inhaltsorientierte Suche im Internet - HTTPs älterer Bruder. *iX Multiuser Multitasking Magazin*, Band 1, S. 120–127, 1995.

[PFH95] Pfeifer, U.; Fuhr, N.; Huynh, T.: Searching Structured Documents with the Enhanced Retrieval Functionality of freeWAIS-sf and SFgate. *Computer Networks and ISDN Systems*, Band 27, Nr. 6, S. 1027–1036, 1995. http://www.igd.fhg.de/www/www95/papers/47/fwsf/fwsf-dead.html.

[Por98] Porst, B.: Konzeption eines Internet-Abo-Dienstes für die Swing Suchmaschine im Rahmen des Projektes Swing, 1998. Studienarbeit an der Universität Rostock, Fachbereich Informatik, LS DBIS http://wwwdb/Lehre/Arbeiten/sa-porst.ps.gz.

[pyt] Python Language Website. http://www.python.org/.

[SB98] Sander-Beuermann, W.: Schatzsucher - Die Internet-Suchmaschinen der Zukunft. *c't magazin für computertechnik*, Band 13, S. 178–184, 1998.

[SBS98] Sander-Beuermann, W.; Schomburg, M.: Information Retrieval: The Further Development of Meta-Searchengine Technology. In: *Proceedings of the 1998 Internet Summit of the Internet Society*, Juli 1998.

[Sd998] Swing-Domänen, 1998.
Informationen zum Suchdienst Swing
`http://swing.informatik.uni-rostock.de/SWING_dom.html`.

[Sew] Search Engine Watch.
`http://searchenginewatch.com`.

[Sg98] Swing - Durchsuchte Domänen, 1998.
Informationen zum Suchdienst Swing
`http://swing.informatik.uni-rostock.de/cgi-bin/gatherers.cgi`.

[Tec96] Technical discussion of the harvest system, 1996.
`http://harvest.transarc.com/afs/transarc.com/public/trg/Harvest/technical.html`.

[w3c] About the World Wide Web Consortium.
`http://www.w3.org/Consortium/`.

[Web98] Weber, A.: Strukturextraktion von RDBMS-Schemata für WWW-DB-Anbindungen, 1998. Diplomarbeit an der Universität Rostock, Fachbereich Informatik, LS DBIS
`http://wwwdb.informatik.uni-rostock.de/Lehre/Arbeiten/da-weber.ps.gz`.

[xml] Extensible Markup Language (XML) 1.0.
`http://www.w3.org/TR/REC-xml`.

[zds] Z39.50 Profile for Simple Distributed Search and Ranked Retrieval.
`ftp://ftp.loc.gov/pub/z3950/profiles/zdsr.ps`.

[Zo98] Z39.50, 1998.
`http://www.biblio-tech.com/html/z39.50.html`.

[Ztd98] Z39.50 - Technical Details, 1998.
`http://www.biblio-tech.com/html/z39.50_part_2.html`.

Abbildungsverzeichnis

Tabellenverzeichnis

Anhang A

Das SOIF

A.1 Formale Beschreibung

Das *Summary Object Interchange Format* (SOIF) wurde im Harvest-System für den Austausch von Objekten zwischen Gatherer und Broker eingeführt. Es handelt sich um einen Attribut-Wert-Datenstrom, dessen formale Beschreibung durch die folgende Grammatik erfolgt:

SOIF	$\longrightarrow$	OBJECT SOIF | OBJECT
OBJECT	$\longrightarrow$	@ TEMPLATE-TYPE {URL ATTRIBUTE-LIST}
ATTRIBUTE-LIST	$\longrightarrow$	ATTRIBUTE ATTRIBUTE-LIST | ATTRIBUTE
ATTRIBUTE	$\longrightarrow$	IDENTIFIER {VALUE-SIZE} DELIMITER VALUE
TEMPLATE-TYPE	$\longrightarrow$	Alpha-Numeric-String
IDENTIFIER	$\longrightarrow$	Alpha-Numeric-String
VALUE	$\longrightarrow$	Arbitrary-Data
VALUE-SIZE	$\longrightarrow$	Number
DELIMITER	$\longrightarrow$	:<tab>

A.2 Beispiel für das SOIF

Anhand des folgenden Beispieles wird die obige Definition veranschaulicht:

```
@File{ http://servername/document.html
File-Size{4}: 3332
Type{4}: HTML
Title{11}: My Homepage
URL-References{66}: http://servername/document2.html
ftp://servername2/directory/file.zip
}
```

Im Beispiel besteht der Datenstrom aus einem einzigen Objekt, das vom Typ File und unter der URL `http://servername/document.html` gespeichert ist. Diesem

Objekt sind die Attribute `File-Size, Type, Title` und `URL-References` zugeordnet. Das Attribut `File-Size` ist mit dem 4 Byte großen Wert `3332` belegt. Die Trennung von Attribut und Wert erfolgt durch die Trennzeichen Doppelpunkt <tab>. Die den Attributen zugeordneten Werte können ein beliebiges Format besitzen; durch die Größenangabe {4} ist immer gewährleistet, daß der Beginn des nächsten Attributes erkannt wird.

A.3 Beispiel für ein vom Gatherer generiertes SOIF Objekt

Der Gatherer erzeugt zu jedem bearbeiteten Dokument ein SOIF Objekt, in dem Informationen über das Dokument (Typ, Größe, Prüfsumme[1], ...), Informationen zum Inhalt des Dokumentes (Autor, Description, Keywords, ...) und Informationen über den Gatherer (Name, Host, Refresh-Rate, ...), der das Dokument bearbeitet hat, erfaßt sind. In [HSW96] ist eine Liste der allgemein verwendeten Attributnamen verfügbar. Ein kurzes Beispiel schließt diese Einführung in das SOIF ab.

```
@FILE { http://wwwdb.informatik.uni-rostock.de/Lehre/fs9798.html
update-time{9}: 906761545
description{85}: Das Forschungsseminar des
+Lehrstuhls Datenbank- und Informationssysteme findet jeden
last-modification-time{9}:      886934255
time-to-live{7}:        2419200
refresh-rate{6}:        604800
gatherer-name{24}:      WWW-Server Volltextsuche
gatherer-host{5}:       hades
gatherer-version{6}:    1.5.19
type{4}:        HTML
file-size{4}:   4393
md5{32}:        24b459785cbcd8224d29dc9588900d31
body{90}:     Universitaet Rostock, Fachbereich Informatik,
Lehrstuhl Datenbank- und Informationssysteme
headings{72}:   Forschungsseminar des Institut fr
+Praktische Informatik, Lehrstuhl DBIS
keywords{42}: datenbank dbis fachbereich html informatik
title{29}:      Forschungsseminar des LS DBIS
url-references{37}:     http://www.informatik.uni-rostock.de/
}
```

[1] Die Prüfsummenberechnung erfolgt mit Hilfe des MD5-Algorithmus (RFC 1321 [md5]), der aus dem Inhalt des Dokumentes einen 16 Byte Schlüssel generiert. Ziel der Prüfsummenberechnung ist die Erkennung von Änderungen im Dokument.

Anhang B

Beispiel für die Integration einer Datenbank

B.1 Der neue Dokument-Typ `ing-db`

Das Beispieldokument `hotel.ing-db` hat pro Zeile die Struktur **Tag** "**:**" **Value**.

```
URL:          http://swing.informatik.uni-rostock.de/hotel-db.html
Description:  Hotel-Datenbank
Keywords:     Hotel
Database:     hotel
Table:        saison
Table:        ortsbeschreibung
Attribute:    ort
Attribute:    geo_region
Table:        zimmerart
Attribute:    zimmername
```

Durch den Summarizer werden die Tags wie folgt interpretiert:

- **URL:** Verweis auf den URL, unter dem das Suchformular verfügbar ist
- **Description:** Kurzbeschreibung des Inhaltes
- **Keywords:** Stichwörter, die den Inhalt des Dokumentes charakterisieren
- **Database:** Name der Datenbank, über die Daten bereitgestellt werden sollen
- **Table:** Ausgabe der Namen aller Attribute aus der sich die Relation zusammensetzt
- **Attribute:** Ausgabe aller Attributwerte dieses Attributes; das Attribut ist in der zuletzt angegebenen Relation (siehe Table) enthalten

B.2 Ausgabe des Summarizers

Der Summarizer generiert daraus das folgende SOIF-Fragment[1], in dem Attributnamen und Attributwerte der Hoteldatenbank angegeben sind:

```
description{15}:        Hotel-Datenbank
keywords{5}:    Hotel
db_hotel{119}:  zimmerlink overview feature_name features
+hotel_min_max charakteristik zimmerart ortsbeschreibung zimmer
saison preise
rel_saison{37}: hotel_id saison_id zeit_bis zeit_von
rel_ortsbeschreibung{68}:       geo_region kult_region ort
+polit_region tour_region x_koord y_koord
ort{214}:       Ahrenshoop Bad Doberan Bentwisch Diedrichshagen
+Graal-M&uuml;ritz Ostseebad K&uuml;hlungsborn Ostseebad Nienhagen
Rostock Rostock-Dierkow Rostock-Markgrafenheide
+Rostock-Warnem&uuml;nde Warnem&uuml;nde Warnemuende
geo_region{48}: Bad Doberan Fischland Ribnitz-Damgarten Rostock
rel_zimmerart{36}:      zimmer_typ zimmername zimmername_uk
zimmername{74}: Appartement Doppelzimmer Dreibettzimmer Einzelzimmer
+Suite Zweibettzimmer
```

B.3 Ausschnitt aus dem Index

Nach dem Mapping und der Anreicherung mit weiteren Informationen enthält der Index folgenden Eintrag[1]:

```
@FILE { http://swing.informatik.uni-rostock.de/hotel-db.html
Update-Time{9}: 906540022
Last-Modification-Time{9}:      906542247
Time-to-Live{7}:        2419200
Refresh-Rate{6}:        604800
Gatherer-Name{24}:      WWW-Server Volltextsuche
Gatherer-Host{5}:       hades
Gatherer-Version{3}:    1.0
Type{6}:        Ing-DB
File-Size{3}:   236
MD5{32}:        284a062c9ba20f2c4cf7ae01a6ec3b25
description{15}:        Hotel-Datenbank
keywords{6}:    hotel
```

[1]Mit + werden Zeilenumbrüche dargestellt, die für die Formatierung der Ausgabe in diesem Dokument notwendig sind. Eine weitere optische Aufbereitung der Ausgabe erfolgt nicht.

```
db_hotel{119}:  zimmerlink overview feature_name features
+hotel_min_max charakteristik zimmerart ortsbeschreibung zimmer
saison preise
rel_saison{37}: hotel_id saison_id zeit_bis zeit_von
rel_ortsbeschreibung{68}:       geo_region kult_region ort
+polit_region tour_region x_koord y_koord
ort{214}:       Ahrenshoop Bad Doberan Bentwisch Diedrichshagen
+Graal-M&uuml;ritz Ostseebad K&uuml;hlungsborn Ostseebad Nienhagen
Rostock Rostock-Dierkow Rostock-Markgrafenheide
+Rostock-Warnem&uuml;nde Warnem&uuml;nde Warnemuende
geo_region{48}: Bad Doberan Fischland Ribnitz-Damgarten Rostock
rel_zimmerart{36}:      zimmer_typ zimmername zimmername_uk
zimmername{74}: Appartement Doppelzimmer Dreibettzimmer Einzelzimmer
+Suite Zweibettzimmer
}
```