Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Thesis Outline

Raoul Pascal Pein

Suchen und Finden im Collaborative Workspace

Fakultät Technik und Informatik
Studiendepartment Informatik

Thema der Thesis Outline
Suchen und Finden im Collaborative Workspace

Stichworte
Bildsuche, Indexierung, Collaborative Workspace

Kurzzusammenfassung
Die meisten derzeit verfügbaren Retrievalsysteme basieren auf einer einzigen Technik, mit der Informationen aus Texten oder auch Bildern gewonnen und verglichen werden. Insbesondere bei Bildern besitzt jede Technik sowohl Vor- als auch Nachteile in Bezug auf die Ergebnisqualität.Im Folgenden wird ein Konzept vorgestellt, das mehrere verschiedene Techniken auf einfache Weise in ein einziges (Image) Retrieval System vereinigen soll. Es wird angenommen, dass dieser Ansatz geeignet ist, insbesondere content-based image retrieval Systeme im Allgemeinen zu verbessern.

Inhaltsverzeichnis

1 Einführung

1.1 Motivation

Mittlerweile haben Unternehmen und selbst Privatpersonen die Möglichkeit, unüberschaubar große Datenmengen zu digital zu archivieren. Dies erschwert das Wiederfinden von Dokumenten aller Art immer mehr. Sobald der Aufwand für das Finden eines Dokumentes größer ist, als das Dokument neu zu erstellen, sind Ansätze zur Verbesserung der Situation erforderlich.

Im Rahmen der „Collaborative Workspace“ Metapher taucht dieses Problem ebenfalls auf. Wenn mehrere Nutzer gleichzeitig auf einem gemeinsamen und hochdynamischen Informationsbestand arbeiten, ist es wünschenswert, dass Jeder bequemen Zugriff auf alle benötigten Dokumente erhält. In diesem Szenario erscheint es hilfreich, über eine flexible Suchfunktionalität zu verfügen.

Diese Arbeit legt ihren Schwerpunkt auf *Content based Image Retrieval* (CBIR), bezieht aber auch Erkenntnisse aus verwandten Retrievalszenarien ein.

1.2 Ansätze für Suchprogramme

Suchprogramme können auf viele verschiedene Weisen realisiert werden. Eine entscheidende Rolle spielen hierbei der Anwendungszweck und die Art der zu durchsuchenden Daten.

Es besteht eine Vielfalt an Datentypen wie Texten, Zeichnungen, Bildern, Multimedia oder sonstigen anwendungsspezifischen Binärdaten. Suchmaschinen müssen ein enormes Spektrum an Umgebungen abdecken. Im einfachsten Fall sitzt ein einzelner Nutzer an einem Gerät. Durch das Internet ist es aber genauso gut denkbar, dass beliebig viele Nutzer gleichzeitig in einem riesigen Suchraum wie dem Internet parallel die selbe Suchmaschine verwenden.

Die Verfügbarkeit der Daten kann problematisch sein. Firmen nutzen beispielsweise verschiedene Sicherheitsstufen, um ihre internen Informationen vor unbefugtem Zugriff zu schützen. Eine Suchmaschine darf also nur den Bestand an Daten durchsuchen, der für den jeweiligen Benutzer zugänglich ist. Untersuchungen dieser Art werden im Folgenden nicht behandelt, aber im Hinblick auf die kommerzielle Nutzung einer Suchmaschine hat dieses Kriterium eine nicht zu unterschätzende Bedeutung.

Textbasiert Der „klassische“ Ansatz, der in den meisten existierenden Suchmaschinen realisiert wird, ist die Textsuche. Sehr viele Dokumente enthalten geschriebenen Text. Dadurch ist es möglich, diese Dateien mit Suchstrings zu durchsuchen.

Neben direkten Vergleichen zweier Strings können beliebig komplexe Algorithmen eine große Flexibilität und Fehlertoleranz realisieren. Im einfachsten Fall sind es Dinge wie das Ignorieren von Groß-/Kleinschreibung. Um Rechtschreibfehler oder andere Störungen zu

beseitigen, können beispielsweise phonetische Algorithmen wie Soundex (1) angewendet werden, die eher unwichtige oder störende Details aus der Suche entfernen.

Diese Art der Suche ist am weitesten verbreitet und wird auf verschiedenste Weise immer weiter optimiert. Im Web sind es Dienste wie Google (5) und für Einzelpersonen oder Gruppen existieren beispielsweise Google Desktop Search (5) oder Beagle Desktop Search (11).

Kataloge Eine andere, alt bewährte Methode ist das Katalogisieren von Dokumenten. Dazu wird eine übergeordnete Struktur aufgebaut, in die jeweils die passenden Dokumente einsortiert werden. Bibliotheken sortieren beispielsweise Bücher nach Genre oder Fachrichtung. Dem Suchenden wird dann nur noch die Metastruktur vorgelegt, die überschaubar gegliedert ist. Jede dieser Kategorien enthält dann einen kleinen, aber relevanten Ausschnitt, in dem das gewünschte Dokument enthalten sein sollte.

Einige Suchmaschinen im Internet bieten ebenfalls derartige Kataloge, die vorgefertigte Suchergebnisse repräsentieren (z.B. (14)). Für eine gute Qualität ist hier eine gewisse Menge an Handarbeit notwendig, da eine hochwertige Kategorisierung nicht vollständig maschinell durchgeführt werden kann. Diese Angebote sind in letzter Zeit schwieriger auszumachen als noch vor einigen Jahren. Oft sind derartige Angebote auch massiv mit Werbung durchsetzt.

Semantik Deutlich anspruchsvoller sind die Vorgehensweisen bei der Erfassung der Semantik. Eine relativ einfache Variante sind die TopicMaps, die beispielsweise von Christensen (2) verwendet werden. Hier werden nicht die Strings selber, sondern deren Bedeutung für das Matching genutzt. Auf diese Weise sollen Mehrdeutigkeiten verringert und gleichzeitig ein zusammenhängender Kontext geschaffen werden.

Im Bereich Semantic Web haben sich im Internet bislang noch keine Suchmaschinen deutlich hervorgehoben. Für eine sinnvolle Nutzung müssen vorher eine einheitliche Dokumentenstruktur und eine weiträumige Nutzung der Semantic Web Technologien etabliert werden.

Krötzsch (9) und Fawzi (4) beschreiben eine interessante Idee, um die starren logischen Formalismen des bisherigen Semantic Web zu umgehen. Sie beruht auf der Annahme, dass das weltweit gesammelte Wissen in Wikipedia bereits besser strukturiert und gleichzeitig umfangreicher ist, als es auf anderen Wegen möglich ist.

Ortsbezogen Mit dem Aufkommen von Online-Kartendiensten wird auch die Suche über Koordianten interessant. Ein Beispiel ist Google Maps (6), welches ermöglicht, Dienstleistungen bezogen auf ihren realen Standort zu finden.

Multimedia Die Suche von Multimediadokumenten ist derzeit ein großes Forschungsfeld. Multimedia unterscheidet sich in den Suchanforderungen wesentlich von einfachen Texten. Das Datenvolumen ist ungleich höher und gleichzeitig finden sich darin keine wiederkehrenden Konstrukte wie Wörter oder Sätze, die für Menschen eine hohe Aussagekraft besitzen. Stattdessen spielen hier abstrake Dinge wie Farbräume, Histogramme, Wavelets oder zeitliche Abfolgen eine Rolle. Diese lassen sich in der Regel nicht über einfache Suchstrings formulieren, sondern bedürfen eines höheren Aufwandes. Zudem lassen diese Daten oft nur eine ungefähre Berechnung der Ähnlichkeit zu, ohne klare Grenzen zwischen „Treffer“ und „Irrelevant“ zu ziehen.

Weit verbreitete Web-Suchmaschinen bieten meist nur Schlagwortsuche oder Kategorien an. Der umfangreiche „BIG Search Engine Index“ (7) kennt von insgesamt 910 verschiedenen Systemen derzeit nur 17 verschiedene webbasierte Bildsuchmaschinen. Keines dieser Systeme bietet CBIR-Funktionalität. Einige der „Suchmaschinen“ stellen sich sogar als sortierte Sammlung von Bildarchiven heraus.

Beispiele für kommerzielles Multimedia Retrieval sind „Excalibur visual retrievalware“ (3) oder „IBM Marvel“ (13). In der Forschung sind Systeme wie „SIMBA“ (12) der Universität Freiburg oder „PictureFinder“ (10) der Universität Bremen zu finden.

1.3 Anforderungen

Die oben genannten Herangehensweisen verdeutlichen, dass ein Retrievalsystem für einen Collaborative Workspace diverse Facetten abdecken muss. Schliesslich sollen hier sowohl Texte als auch Multimediadaten verarbeitet werden.

Da im Bereich textbasierter Suche bereits viele nutzbare Werkzeuge existieren, liegt der Schwerpunkt dieser Arbeit auf CBIR. Die wichtigsten Anforderungen an ein Retrievalsystem sind folgende:

Precision/Recall Die Suchergebnisse müssen eine hohe Qualität erzielen. Zum Einen müssen die Ergbnisse einen deutlichen Bezug zur gestellten Anfrage haben. Zu viele irrelevante Dokumente mindern die Übersichtlichkeit. Zum Anderen muss sichergestellt sein, dass möglichst alle relevanten Dokumente im Ergebnis erschienen. Tauchen sie bei der Suche nirgendwo auf, sind sie praktisch nicht existent.

An dieser Stelle hat das CBIR einen deutlichen Nachteil gegenüber anderen Techniken. Wie in Abschnitt 1.1 dargestellt, werden hier Ähnlichkeiten berechnet. Deshalb können Anfragen in der Regel nur unscharf formuliert werden und die Abgrenzung der Ergebnismenge vom Rest des Bestandes ist nicht trivial.

Geschwindigkeit Die Zeitersparnis durch eine Suche gegenüber anderer Navigation muss deutlich spürbar sein. Die Antwortzeiten sollten möglichst im Sekundenbereich liegen. Lange Antwortzeiten können den Benutzer dazu bringen, Dokumente auf anderem Wege zu finden.

Skalierbarkeit Mit der Forderung nach Geschwindigkeit hängt direkt die Skalierbarkeit zusammen. Eine Suche in einem großen Datenbestand sollte nicht wesentlich länger dauern, als in einem kleinen. Andernfalls besteht die Gefahr, dass eigentlich vorhandene Dokumente durch Nichtverwendung der Suche ignoriert werden.

Hinzu kommt, dass bei einem großen Datenbestand auch der „Verschmutzungsgrad“ an irrelevanten Daten zunimmt. Eine gute Ergebnisqualität wird mit zunehmender Größe also immer wichtiger.

Wartbarkeit Die Administration der Indexdaten darf den Nutzen einer Suche nicht durch hohen Aufwand wieder zunichte machen. Die Indexerstellung sollte möglichst stark automatisiert sein.

Erweiterbarkeit Das System soll nicht auf bestimmte Datentypen beschränkt sein. Im Prinzip kann jede vorhandene Datei auf irgendeine Weise in den Index aufgenommen werden. Voraussetzung ist hierbei, dass ein datentypspezifischer Algorithmus geschrieben werden kann, der suchbare Daten extrahiert.

Plattformunabhängigkeit Da gerade im Szenario des „Collaborative Workspace“ viele verschiedene Geräte zum Einsatz kommen können, ist eine plattformunabhängige Architektur wünschenswert.

Ergonomie Alle bisher genannten Anforderungen münden zwangsläufig in der Ergonomie. Oberstes Ziel ist die Bereitstellung eines Dienstes, der das Wiederfinden von Dateien auf bequeme Weise ermöglicht. Der Mehrwert dieses Dienstes muss deutlich erkennbar sein. Gleichzeitig muss die Bedienung schlank gehalten werden, um Verwirrung oder eine aufwändige Konfiguration der Suchparameter zu vermeiden.

Auf diese Weise soll der Anwender einen Vorteil in der Software erkennen und sie im täglichen Gebrauch einsetzen.

2 Eigener Ansatz

Dieser Teil stellt den Prototypen „Golden Retriever“ vor, der die Anforderungen aus 1.3 realisieren soll.

2.1 Vision

Das angestrebte Ziel ist ein Dienst, der verschiedenste Anfragen ermöglicht und Dokumente beliebigen Typs in kurzer Zeit liefert. Die Unterstützung von Suchkriterien und Filtern soll frei erweiterbar sein. Beispiele für Suchkriterien (im Folgenden auch *Aspekte* genannt) sind Schlagworte, Kategorie oder Bildinhalte. Die Anfragen sollen direkt oder über Beispiele (z.B. Zeichnungen) formulierbar sein.

Die Verwaltung des Systems soll möglichst einfach gehalten werden. Was bei der Indexierung nicht automatisierbar ist, soll von allen Nutzern des Systems nachträglich editiert werden können.

2.2 Schnittstellen

Die API basiert auf WebServices. Hierbei ist angestrebt, vorerst möglichst zustandsfrei zu arbeiten. Jede Anfrage ist somit völlig eigenständig und unabhängig von den anderen. Das Standard-Rückgabeformat ist XML-basiert. Zum Teil sind auch andere Formate sinnvoll, wie beispielsweise bei der Übertragung von Bilddaten.

Die zentrale Funktionalität wird von *getRanking* geboten. Als Parameter sind die externe URL oder interne ID einer bereits vorhandenen Datei, sowie Schwellwerte für die Begrenzung des Suchraums möglich. Zusätzlich können beliebige implementierte Aspekte inklusive Gewichtung und Schwellwerten übergeben werden. Als Ergebnis wird eine Liste mit IDs zurückgegeben, die wahlweise auch Zusatzinformationen wie die berechnete Ähnlichkeit enthalten kann.

Um die benötigten Daten für die Anfrage zu erhalten, können die Daten clientseitig direkt in einen Parameterstring konvertiert werden. Alternativ bietet *calculateFeatureVector* die Möglichkeit, aus einer URL diesen String serverseitig zu berechnen. Eine zufällige Auswahl an Dateien wird über *getRandomImages* angeboten.

Für die Darstellung werden des weiteren Zusatzinformationen benötigt. Diese sind über Befehle wie *getImage*, *getImageData*, *getItemData* und *getFeatureVectorInfoMap* verfügbar.

Das Hinzufügen neuer Informationen geschieht über *addImage*, *addItem* und *setFeatureVector*.

Der Dienst selbst benötigt Schnittstellen zu der darunterliegenden Persistenzschicht und dem zu integrierenden Beagle (11).

Die Persistenzschicht soll primär über das Dateisystem direkt angesprochen werden. Zusätzliche Dienste wie Clustering und Versionskontrolle sind geplant und benötigen eine gesonderte API.

Der Suchdienst Beagle bietet von sich aus bereits eine nutzbare API. Es ist zu prüfen, ob ein zusätzlicher Notify Service auf einfache Weise integriert werden kann.

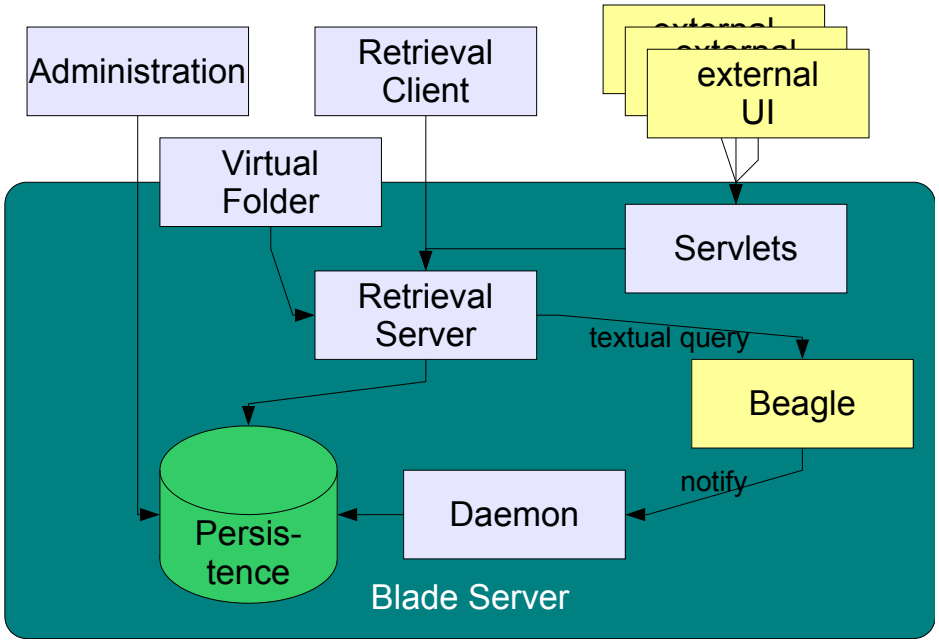

Abbildung 1: Architekturübersicht

2.3 Architektur

Die Architekur des Retrievalsystems richtet sich nach den Besonderheiten beim Content Based Image Retrieval. Dabei wird auf die Erweiterbarkeit und Flexibilität mehr Wert gelegt, als auf die Geschwindigkeit.

Das Programm unterscheidet bei der Erstellung der Ergebnismenge nicht strikt nach „Treffer“ und „Irrelevant“, wie bereits in Sektion 1.1 beschrieben. Statt dessen wird eine Ähnlichkeit zwischen Anfrage und den durchsuchten Dateien berechnet. Diese Ähnlichkeit kann im Grunde beliebig komplex definiert sein. Entscheidend ist nur, dass sie auf einen eindimensionalen Wert im Bereich [0.0, 1.0] heruntergebrochen werden kann, wobei 1.0 für die Identität steht und kleinere Werte für wachsende Unterschiede. Mit diesem Ähnlichkeitswert wird ein aspektspezifisches Teilranking erstellt. Mehrere Teilrankings lassen sich dann zu einem Gesamtranking zusammenführen. Zuerst wird nur die gewichtete Summe der Teilähnlichkeiten berechnet. Hier sind auch andere Verfahren denkbar, die beispielsweise Vereinigungs- oder Schnittmengen bilden.

Das Zusammenspiel der einzelnen Komponenten ist in Abbildung 1 dargestellt. Den Kern bilden die Komponenten *RetrievalServer* und die *Persistenz*. Der Server nimmt die eigentlichen Berechnungen auf Basis der Daten in der Persistenzschicht vor. Im Rahmen der Diplomarbeit wurden diese Teile zusammen mit dem *RetrievalClient* und dem *Administration-*

Tool realisiert. Derzeit wird die Persistenz über eine MySQL-Datenbank realisiert. Geplant ist eine parallele Realisierung, die direkt auf dem Dateisystem arbeitet.

Vorbereitend auf die Master-Thesis wurde im Projekt die Servlet-Schnittstelle für die Nutzung im Labor realisiert und der bestehende Code größtenteils überarbeitet.

Im Folgenden ist es geplant, einen Hintergrundprozess zu integrieren, der automatisch neue oder veränderte Dokumente erkennt und in den Index mit aufnimmt. An dieser Stelle soll der bestehende iNotify-Mechanismus von Beagle (11) genutzt werden. Die Verwendung von Beagle hat den weiteren Vorteil, dass dieser rein textbasierte Suchdienst parametrisiert vom RetrievalServer aufgerufen werden kann. Dadurch wird eine Vorabfilterung des eigenen Index erzielt, so dass der Suchraum deutlich eingeschränkt wird.

Als optionale Erweiterung ist der *VirtualFolder* Dienst vorgesehen. Dieser soll auf Dateisystemebene transparenten Zugriff auf Suchergebnisse bieten. Dazu wird ein Verzeichnis mit einer abspeicherbaren Anfrage erstellt. Mit Hilfe dieser Anfrage wird nun das Verzeichnis mit Referenzen auf die tatsächlichen Dateien dynamisch gefüllt. Hier erscheint die Verwendung von „harten/symbolischen Links“ aus Unix-Systemen sinnvoll.

Die serverseitigen Komponenten sollen auf einem oder mehreren Blades im Labor lauffähig sein.

2.4 Skalierung

Bei der Skalierung können typische Probleme auftreten.

Derzeit ist die Suchstrategie noch keineswegs optimiert. Eine Suche führt einen kompletten Scan über alle vorhandenen Datensätze durch. Daher steigt der Rankingaufwand steigt linear mit der Größe der Datenbank. Eine Ähnlichkeitssuche lässt sich nicht direkt auf einfache Indexe abbilden. Praktisch jeder Vergleich liefert eine Ähnlichkeit > 0,0. Dadurch existieren keine klaren Grenzen, welches Objekt in die Ergebnismenge gehört. Ein Lösungsansatz besteht darin, den Suchraum im Vorfeld so stark wie möglich einzugrenzen, ohne dabei wichtige Ergebnisse zu verlieren. Dies kann beispielsweise über Clustering, Mehrdimensionale Suchbäume oder „harte“ Filter (Keywords, Kategorien, Tags) realisiert werden. Die rechenaufwendigen Vergleiche müssten dann nur noch über einen kleinen Teilbereich durchgeführt werden.

Weiterhin kann der Index so stark anwachsen, dass die Daten nicht mehr vollständig im Arbeitsspeicher gehalten werden können. Dies hätte eine massive Beeinträchtigung der Antworzeiten zur Folge. Es sind Teiloptimierungen denkbar, indem die Datengenauigkeit gesenkt wird oder nur die am häufigsten genutzten Aspekte komplett im Speicher gehalten werden.

Die Indexerstellung ist gerade bei hochauflösenden Bildern teuer, weil die genutzen Daten oftmals aus dem gesamten Pixelbild extrahiert werden müssen. An dieser Stelle könnte eine einmalige Vorskalierung auf eine geringere Größe helfen, gerade bei vielen zu extrahierenden Aspekten Rechenzeit zu sparen.

Das Thema Load Balancing wird an dieser Stelle ebenfalls interessant, da im Labor mehrere Server zur Verfügung stehen. Diese könnten sich die gegebenenfalls zeintintensive Erstellung von Subrankings aufteilen und parallel bearbeiten. Anschließend müssen die Zwischenergebnisse nur noch an zentraler Stelle zusammengefügt werden. Eine verhältnismäßig einfache Implementierung kann beispielsweise über ein Blackboard realisiert werden.

3 Zukünftiges Vorhaben

Die Themen der Master Thesis können in vier Teilbereiche eingeordnet werden:

- Nahtlose Integration des bestehenden Prototypen in das gemeinsame Projekt (Servlets und dynamische Samba-Shares)
- Nutzung von externen Diensten wie Beagle zur Abbildung von ergänzenden Metainformationen und Semantik
- Ermöglichung von manueller Annotation zur Erweiterung/Verfeinerung der automatisch erstellten Indexdaten
- Evaluation des erhofften Zusatznutzens von kombinierter Suche und manueller Annotation

Die ersten zwei Punkte wurden bereits in den Abschnitten 2.2 und 2.3 kurz beschrieben. Eine genauere Beschreibung der Schittstelle für manuelle Annotation ist bisher noch nicht verfügbar. Dies wird sich im Laufe der Zeit ergeben, sobald das parallele Projekt zur Visualisierung des Dienstes (P. Roßberger) Formen annimmt. Dabei spielen insbesondere die verwendeten Eingabegeräte eine wichtige Rolle.

3.1 Evaluierung

Ein wichtiger Teil der Master-Thesis wird die Evaluierung des fertigen Systems sein. Hiebei liegt das Hauptaugenmerk auf der Qualität der angeboteten Suchergebnisse und die verschiedenen Wege, auf denen sie erzielt wurden. Es wird erwartet, dass die Suche über einzelne Aspekte im Mittel schlechtere Ergebnisse erzielt als die Suche über kombinierte Aspeke. Außerdem soll die Bereitschft der Nutzer ermittelt werden, verschieden komplexe Anfragen selbst zu formulieren.

Qualität Im Folgenden wird ein möglicher Versuchsablauf zur Überprüfung der subjektiven Ergebnisqualität beschrieben. Für dieses Vorgehen werden möglichst viele Versuchspersonen benötigt, die jeweils eine Reihe von Suchergebnissen im Vergleich zu einer vordefinierten Anfrage beurteilen. Dazu werden im Vorfeld die zu untersuchenden Anfragen festgelegt.

Zum Einen sind eine Reihe Bilder und Annotationen zu definieren. Für diese ist eine Anzahl passender Bilder im Datenbestand vorhanden, die im Optimalfall auch im Ergebnis auftauchen sollen. Zum Anderen wird eine Reihe von zufällig ausgewählten Bildern berechnet.

Zusätzlich zu diesen Daten wird festgelegt, auf welche Weise die einzelnen Aspekte miteinander kombiniert werden. Im einfachsten Fall wird in der Suche nur ein einzelner Aspekt genutzt, ansonsten werden mehrere mit verschiedenen Gewichtungen kombiniert.

Der Ablauf für eine einzelne Testperson wird relativ starr gehalten. Dadurch soll es ermöglicht werden, viele Probanden mit unterschiedlichem Kenntnisstand mit dem System zu konfrontieren. Der geplante Ablauf einer Sitzung ist:

1. Das System wählt automatisch eine vordefinierte Anfrage aus oder erstellt eine zufällige
2. Der Proband erfährt optional, nach welchen Kriterien gesucht wird
3. Die Ergebnismenge wird dem Probanden angezeigt
4. Der Proband selektiert alle Ergebnisse, die er als ähnlich ansieht
5. Nach einer Bestätigung wird der Vorgang N mal mit anderen Daten wiederholt

Die erhaltenen Daten sollen bei ausreichend vielen Testdurchläufen statistisch ausgewertet werden. Es wird erwartet, dass Zusammenhänge zwischen Ergebnisqualität und verwendeter Anfrage erkennbar sind.

Bedienbarkeit Die Untersuchung der Bedienbarkeit erfordert mehr Aufwand bei der Versuchsvorbereitung. Eine Untersuchung über die effiziente Nutzung eines Ergonomie-Labors ist von Keseling (8) vorgenommen worden.

Die Benutzungsschnittstelle ist bei einem Suchprogramm in der Regel relativ einfach gehalten. Normalerweise werden nur die Anfrage erstellt und die Ergebnisse möglichst übersichtlich dargestellt. Die in dieser Arbeit offen angelegte Architektur erfordert eine komplexe und flexible Bedienoberfläche. Daher besteht die Gefahr, dass die Komplexität der Bedienung überproportional zum eigentlichen Nutzen der Software steigt.

Da neben der eigentlichen Suche auch die manuelle Annotation der Daten möglich sein soll, ist hier ein weiterer Punkt, an dem die Bedienbarkeit von Bedeutung ist.

Diese Untersuchung wird voraussichtlich zusammen mit P. Roßberger vorgenommen werden, der eine grafische Oberfläche erstellen wird. Diese wird für die Nutzung im Collaborative Workspace Labor optimiert.

3.2 Risiken

- Das System wird vom Benutzer nicht akzeptiert
- Evaluierung von CBIR generell schwierig, da es keine Referenzprojekte gibt
- Samba-Shares möglicherweise extrem aufwändig zu implementieren
- Offen zugängliche Systeme sind anfällig gegen Angriffe (z.B. Wikis)

Es kann sich durchaus herausstellen, dass das fertige System nicht so gut benutzbar ist, wie ursprünglich erwartet. In diesem Fall soll zumindest die Frage geklärt werden, warum die Akzeptanz so niedrig ist. Die Evaluierung der Ergebnisqualität muss sorgfältig vorbereitet werden, damit im Vorfeld aussagekräftige Resultate sichergestellt werden können. Die Priorität bei der Implementierung des Prototypen liegt auf der Seite von Bedienbarkeit und guter Ergebnisqualität. Die Optimierung der Antwortzeiten kann in einem späteren Schritt geschehen, so lange die Antwortzeiten bei einer Menge von ca. 1000 verwalteten Dateien akzeptabel bleiben.

Technische Risiken wie die Implementierung von dynamischen Samba-Shares oder einem Load-Balancing sind nicht drekt abzusehen. Daher werden diese Ziele nur optional realisiert.

Die geplante verteilte Annotation vieler Anwender parallel kann sich ebenfalls als problematisch herausstellen. Sollten sich die Hoffnungen bezüglich einer kooperativen Nutzung des Angebotes nicht erfüllen, wird der Schreibzugriff auf eine ausgewählte Gruppe von Administratoren beschränkt.

Literatur

[1] The soundex indexing system. Available from: `http://www.archives.gov/publications/general-info-leaflets/55.html`.

[2] Andreas Christensen. Semantische Anreicherung von Suchanfragen auf Basis von Topic Maps. Diplomarbeit.

[3] Excalibur visual retrievalware. Available from: `http://www.excalib.com/`.

[4] Marc Fawzi. Wikipedia 3.0: The end of google? Available from: `http://evolvingtrends.wordpress.com/2006/06/26/wikipedia-30-the-end-of-google/`.

[5] Google, 2007. Available from: `http://www.google.com/`.

[6] Google Maps, 2007. Available from: `http://maps.google.com/`.

[7] Image search engines. Available from: `http://www.search-engine-index.co.uk/Images_Search/`.

[8] Sebastian Keseling. Durchführung eines Usability-Tests zur Verbesserung der Untersuchungsmethodik, September 2004. Diplomarbeit.

[9] Markus Krötzsch, Denny Vrandečić, and Max Völkel. Wikipedia and the semantic web - the missing links. In *Proceedings of the WikiMania2005*, 2005. Available from: `http://www.aifb.uni-karlsruhe.de/WBS/mak/pub/wikimania.pdf`.

[10] A. Miene, Th. Hermes, and G.T. Ioannidis. Graphical Image Retrieval with PictureFinder. In *DELOS Workshop on Multimedia Contents in Digital Libraries*, 2003.

[11] Joe Shaw. Beagle desktop search, 2006. Available from: `http://beagle-project.org/Main_Page`.

[12] Sven Siggelkow, Marc Schael, and Hans Burkhardt. SIMBA - Search IMages By Appearance. In *Pattern Recognition: 23rd DAGM Symposium, Munich, Germany, September 12-14, 2001. Proceedings*, volume 2191/2001, page 9, 2001.

[13] John R. Smith. MARVEL: Multimedia Analysis and Retrieval System. Technical report, Intelligent Information Management Dept., IBM T. J. Watson Research Center, 19 Skyline Drive, Hawthorne, NY 10532 USA, 2004.

[14] Yahoo! Search - Directory Search, 2007. Available from: `http://search.yahoo.com/dir`.