HTWK Leipzig

Magisterarbeit

Konzeption eines personalisierenden Suchagenten für das WWW

Vorgelegt dem Fachbereich Informatik, Mathematik und Naturwissenschaften von

Stefan Kluge

KlugeStefan@web.de

Krondorfer Straße 119
06766 Wolfen
Matrikelnummer: 21957

eingereicht am 17.7.2003

Erstgutachter: Prof. Dr. Klaus Hänßgen
Zweitgutachter: Prof. Dr. Frank Jäger

Inhaltsverzeichnis

1. Einleitung 1-4
2. Grundlagen 2-5
2.1. Begriff: Information 2-5
2.1.1. Geschichte der Informationswissenschaft 2-5
2.1.2. Zweckmäßige Betrachtung des Informationsbegriffs 2-6
2.2. Begriff: Softwareagent 2-13
2.2.1. Definition: Agent 2-13
2.2.2. Eigenschaften von Softwareagenten 2-14
2.2.3. Klassifikation von Softwareagenten 2-21
2.2.4. Softwareagenten als WWW-Assistenten 2-35
2.2.5. Herausforderungen an Softwareagenten 2-40
2.2.6. Ausblick zu den Softwareagenten 2-41
2.3. Begriff Personalisierung 2-45
2.3.1. Sicherheitsaspekte und Datenschutz 2-46
2.3.2. Überblick über Personalisierungstechniken 2-50
3. Konzeption 3-54
3.1. Machbarkeit 3-54
3.2. Anforderungen 3-57
3.3. Systementwurf 3-64
3.3.1. Phase 1 – Bestimmung des Informationsbedürfnisses 3-65
3.3.2. Phase 2 – Informationsgewinnung und Personalisierung 3-77
3.3.3. Phase 3 – Informationsausgabe und Lernmethoden 3-85
4. Zusammenfassung und Ausblick 4-93
5. Literatur und Verzeichnisse 5-95

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benutzt und die den verwendeten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Leipzig, Stefan Kluge

1. Einleitung

Das World Wide Web (WWW) breitete sich mit einer beispiellosen Geschwindigkeit aus und damit auch die Menge an Informationen, die über das WWW verfügbar sind. Gleichzeitig haben immer mehr Anwender Zugang zu dieser Informationsquelle, während die Vertrautheit mit den zunehmend komplexeren technischen Grundlagen dieses Mediums einem immer kleineren Anteil von Experten vorbehalten bleibt.

Das Bedürfnis an Werkzeugen zur Informationssuche im Internet ist groß. In dieser Arbeit werden zwei Grundkonzepte erläutert, die nach Meinung des Autors bei der Entwicklung fortgeschrittener Suchwerkzeuge zukünftig eine zentrale Rolle spielen werden: Softwareagenten und Personalisierungskonzepte.

Auf die Grundlagen soll dabei besonders eingegangen werden, um ein Verständnis der Sachlage und des aktuellen Stands der Entwicklung zu ermöglichen sowie Überlegungen anzuregen, die über die in dieser Arbeit beschriebenen Verfahren hinausgehen.

Die Konzeption dient vor allem der Veranschaulichung der zum Teil abstrakten Gegebenheiten und soll deutlich machen, dass die Entwicklung fortgeschrittener Suchwerkzeuge mit den in dieser Arbeit beschriebenen Konzepten möglich ist.

2. Grundlagen

2.1. Begriff: Information

Informationen sind das Medium, in dem Suchagenten ihren Aufgaben nachgehen. Da in dieser Arbeit die Konzeption eines Suchagenten erläutert wird, muss zunächst auf den Grundbegriff **Information** eingegangen werden.

2.1.1. Geschichte der Informationswissenschaft

Das Wort Information leitet sich von **informatio** ab, einem Wort, das in der lateinischen Umgangssprache gebraucht wurde. Die in dem Wort enthaltene Wurzel **forma** geht auf Begriffe der griechischen Philosophie zurück (eidos, idea, morphé, typos), die in diesem Zusammenhang von den bedeutenden Philosophen Platon und Aristoteles gebraucht wurden. In der Ideenlehre, dem Kernstück der Philosophie Platons, tauchte das Wort idea erstmals auf. Platon postuliert die **Idee** als abstraktes Ding; konkrete Dinge seien lediglich als Abbildungen dieser a priori existierenden Ideen zu verstehen.

Der Zusammenhang zwischen der Idee und der Information wird in dem bekanntesten Text Platons deutlich, dem Höhlengleichnis:

> *Eine Gruppe von Menschen ist schon von Geburt an in einer Höhle so festgebunden, dass ihr Rücken stets dem Höhleneingang zugewandt ist und sie nur auf die Höhlenwand vor sich blicken können. Alles, was vor dem Höhleneingang vorbeigetragen wird oder daran vorbeigeht, wirft einen Schatten an die Wand. Die Menschen in der Höhle kennen nichts*

anderes, als diese Abbilder der wirklichen Dinge, welche ihnen nicht nur unbekannt, sondern auch unbegreiflich sind.[1]

Platon veranschaulicht eine idealistische, abstrakte Sicht auf Dinge (Stichwort Platonischer Ideenhimmel). Unsere Wahrnehmung wird beschränkt auf die Informationen, die uns erreichen (z.B. über das Licht oder durch den Schall). Der informationstheoretische Ansatz macht die Information von einem Informationsträger abhängig. In der Informationswissenschaft wird die Frage nach der Information ohne Informationsträger (analog zur Platonischen Idee) kontrovers diskutiert.

Die Informationstheorie beschäftigt sich mit den mathematischen und statistischen Grundlagen der Nachrichtenübertragung. Die Forschungen zur Informationstheorie wurden wesentlich von Claude Shannon in den 40er Jahren des 20. Jahrhunderts begründet. Shannon untersuchte die Faktoren, die Einfluss nehmen auf die Übertragung von Nachrichten über nicht-ideale Übertragungskanäle (z.B. „verrauschte" Telefonleitungen). Er prägte auch den Begriff Entropie.

Für eine umfassende interdisziplinäre Einführung in den Informationsbegriff sei auf [Capurro, 2003] verwiesen.

2.1.2. Zweckmäßige Betrachtung des Informationsbegriffs

Der Begriff Information hängt eng mit der Entropie zusammen. Die Entropie gibt den Informationsgehalt (den Gehalt an Neuigkeiten) in einer Variable an. Der Inhalt einer Variable kann in der Informatik als Datum bezeichnet werden, in einem technischeren Kontext als Nachricht. Ein Datum, das stets denselben Wert darstellt (Konstante), ist vorhersehbar und hat somit keinen

[1] aus „Wikipedia – Die freie Enzyklopädie". http://de.wikipedia.org/wiki/H%F6hlengleichnis (20.06.2003)

Informationsgehalt. Seine Entropie ist 0. Ein zufälliges Datum hat eine Entropie von 1. Damit ein System einer Information innerhalb eines bestimmten Kontextes eine Bedeutung zuordnen kann, muss die Entropie dieser Information zwischen 0 und 1 liegen, man spricht in diesem Fall auch von Nutzinformation. Nutzinformationen können in diesem Zusammenhang als Wissen bezeichnet werden. Das Wissen erlaubt es einem System, auf Reize angemessen zu reagieren, angemessen im Sinne der Verfolgung einer Strategie.

Um diese Begriffsfindung zusammenzufassen sollen drei Ebenen zum Begriff Information herangezogen werden, die bereits in [Kluge und Menzel, 2002] im Zusammenhang mit der Personalisierung genannt wurden und in der Wissensverarbeitung verbreitet sind:

Informationsebenen

- **Datum:** kleinste potentielle Informationseinheiten; (Datum in der technischen Informatik: das Bit); syntaktische Ebene
- **Information:** strukturierte Daten mit $0 < H(p) < 1$ (H: Entropie; p: Zufallsfunktion der Daten auf denen die Information basiert); semantische Ebene
- **Wissen:** Information in einem bestimmten Kontext betrachtet; pragmatische Ebene

Man spricht im Allgemeinen auch von den folgenden drei Informationsebenen: Syntax, Semantik und Pragmatik. In Abbildung 1 werden die Beziehungen zwischen den Ebenen vereinfacht dargestellt. Dies dient der Veranschaulichung; tatsächlich lassen sich diese Ebenen nur schwer trennen.

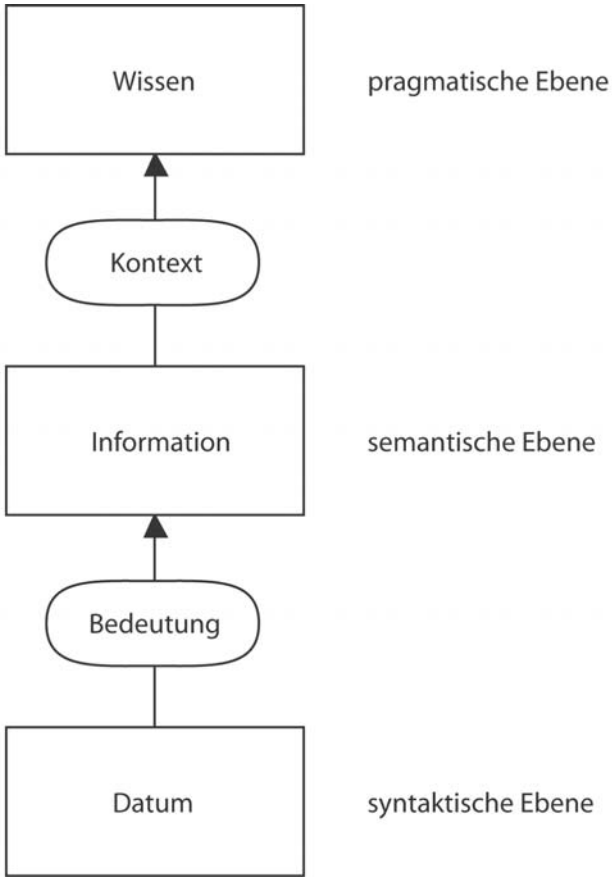

Abbildung 1: drei Informationsebenen und deren Zusammenhang

Die Ebenen sollen anhand eines Beispieles betrachtet werden.

Beispiel zur Veranschaulichung der Informationsebenen

Man stelle sich ein intelligentes Individuum vor: einen Menschen. Dieser Mensch vernimmt ein Geräusch, d.h. einen Schall. Physikalisch gesehen besteht der Schall aus mechanischen Schwingungen, der Schwingungsträger ist die Luft, die aus atomaren Teilchen besteht. Das Geräusch basiert auf der kleinsten Ebene demnach auf Schwingungen der atomaren Teilchen. Die Analogie in der hier beschriebenen Begriffswelt wäre folgende: die atomaren Teilchen können als Daten bezeichnet werden.[2]

[2] Die Annährung an den Begriff Information über die Betrachtung des Informationsträgers hat ihre Wurzeln vor allem in der Nachrichtentechnik.

Aufgrund seiner Erfahrungen misst der Mensch dem Geräusch eine Bedeutung zu. Er stellt fest, dass es sich um das Geräusch eines sich schnell nähernden Autos handelt. Die Schwingung dieser Teilchen, d.h. das Geräusch, wird von dem Menschen dazu in eine Information überführt. Der Träger dieser Information ist die Luft. Der Vorgang, dem Geräusch eine Bedeutung zuzumessen, entspricht der zweiten Ebene unserer Analogie – es handelt sich nun um eine Information: „ein Auto nähert sich schnell".

Um auf Basis dieser Information eine Entscheidung zu treffen (oder aus ihr eine Erkenntnis zu erlangen), setzt der Mensch die Information ins Verhältnis zu seiner Umwelt: da er sich auf der Straße befindet, das Geräusch sich ihm nähert und Autos normalerweise auf der Straße fahren, springt er augenblicklich von der Straße weg. Dies ist die Wissensebene, die ein Bewusstsein voraussetzt, d.h. ein Verständnis über die Beziehung des Individuums zur Umwelt. Wissen hat einen pragmatischeren, weniger abstrakten Charakter als die Information.

Eigenschaften von Informationen

Eine vollständige Auflistung der Eigenschaften von Informationen ist aufgrund der Abstraktheit des Begriffes nicht möglich. Um den Begriff fassbar zu machen, sollen dennoch einige Eigenschaften herausgegriffen werden:

- **Informationsträger:** Was kann als Informationsträger dienen? Prinzipiell kann man sagen, dass jegliche Materie Informationsträger sein kann. Über die Luft werden akustische Informationen übermittelt, über das Licht nimmt man optische Informationen war, Körper können ertasten werden. Ein Stein enthält Informationen über sein Alter, Herkunft etc. Steine können als Informationsträger für von Steinen physikalisch unabhängigen Informationen gebraucht werden: man kann damit Grundstücksgrenzen markieren oder einen Wegesrand kennzeichnen.

Vorausgesetzt, die Informationsempfänger verfügen über das gleiche Wissen, was Steine in diesem Zusammenhang zu bedeuten haben; eine gemeinsame Ontologie.

Sind Informationen auf Materie beschränkt? Viele Philosophen und Metaphysiker vertreten die Auffassung von der immateriellen Idee (wie in Kapitel 2.1.1 beschrieben). Danach können Informationen auch in immateriellen Ideen stecken; ein materieller Informationsträger ist nicht notwendig.

- **Informationsklassen:** Eine allgemeine Klassifikation von Informationen ist nicht möglich. Je nach Anwendungsfall können Klassen jedoch sinnvoll sein:
  - nach Struktur: strukturiert, unstrukturiert
  - nach Wahrnehmung: akustisch, optisch, geruchlich etc.
  - nach Verfügbarkeit: öffentlich, geheim
  - nach Codierung: binär (Rechner), farblich (Ampel) etc.
- **Redundanz:** Ableitbare Informationen sind redundant. Bei einer Informationsübertragung kann eine Information weggelassen werden, wenn die Ableitungsregel dafür bekannt ist. In der Nachrichtentechnik spielt die Redundanz eine wichtige Rolle. Um eine Fehlererkennung oder Fehlertoleranz bei der Übertragung von Informationen über verlustbehaftete Übertragungskanäle zu gewährleisten, werden redundante Informationen bewusst hinzugefügt, z.B. in Form von Prüfsummen.

  Beim Konzipieren von Datenbanken hingegen achtet man in der Regel auf Redundanzfreiheit. Um Inkonsistenzen bei der Manipulation von

Daten innerhalb der Datenbank zu vermeiden, sollten Informationen nach Möglichkeit nur einfach abgelegt werden (aus Performancegründen wird diese Design-Richtlinie gelegentlich ignoriert).

Da materielle Informationsträger verloren gehen können (und somit für den Empfänger die Information verloren geht), werden wichtige Informationen oft bewusst redundant gehalten, z.B. durch Spiegelungen gesamter Datenbanken oder Sicherheitskopien. Auch die Natur sieht solche Schutzmaßnahmen vor: Die Erbinformationen, getragen durch die Desoxyribonukleinsäure (DNA), sind hoch redundant.

- **Kompression:** Eine der häufigsten Aufgaben der Informatik ist die Modellierung reeller Prozesse (Geschäftsprozesse, natürliche Phänomene etc.). Dabei werden Informationen verarbeitet, die zum Teil redundant sind. Da Verarbeitungskapazitäten begrenzt sind, ist es sinnvoll, Redundanz (vorübergehend) zu entfernen. Dafür stehen den Informatikern Kompressionsverfahren zur Verfügung, die in der Regel im Hinblick auf den Informationstyp (z.B. mp3 bei akustischen Informationen) oder die Informationsstruktur (z.B. Run-Length-Encoding bei langen Folgen identischer Daten) entworfen wurden.

- **Qualität:** Um die Qualität einer Information zu beurteilen, muss man diese auf der pragmatischen Ebene betrachten: Eine Information kann in unterschiedlichen Anwendungsfällen eine unterschiedliche Qualität haben, d.h. unterschiedlich hochwertig sein. So ist eine hochwertige Information im Information Retrieval eine, die der gesuchten Information (z.B. repräsentiert durch einen Suchstring) möglichst nahe kommt.

Interessant ist, dass ein Ding, für das Information unterschiedliche Qualität haben kann, ein Verständnis über die Beziehung zu seiner Umwelt haben muss. Wie bereits festgestellt wurde, ist dieses Verständnis (der Kontext) auf der pragmatischen Informationsebene ohnehin notwendig. Im Bezug auf die Entwicklung von Software zum Information Retrieval ist diese Feststellung wichtig. Um die Qualität der erhaltenen Informationen beurteilen zu können, ist also ein Verständnis über den Kontext notwendig, d.h. über die Umgebung des Information Retrieval Systems. Um ein solches Verständnis technisch zu realisieren, kann man sich an der in diesem Kapitel erarbeiteten Begrifflichkeit orientieren. So ist eine Architektur analog zu den Informationsebenen sinnvoll. Auch Informationsträger, Informationsklassen und Redundanz werden eine Rolle spielen.

2.2. Begriff: Softwareagent

Der Begriff **Agent** wird besonders im Zusammenhang mit kommerziellen Systemen oft ungenau verwendet. Welche Eigenschaften Software aufweisen muss, um der, zumindest in der Forschung weitgehend einheitlichen Definition eines Softwareagenten gerecht zu werden, wird in diesem Kapitel erläutert. Es soll ein Blick auf die Geschichte der Softwareagenten geworfen werden, anhand einer Klassifikation auf existierende Konzepte der Softwareagenten eingegangen werden sowie ein Ausblick auf die Entwicklung dieses Forschungsgebietes geben werden.

2.2.1. Definition: Agent

In der Literatur sind viele z.T. stark abweichende Definitionen zu finden. Eine Definition nach [Wooldridge, Jennings, 1995a] lautet wie folgt:

> *"An agent is a computer system, that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives."*

Es ist ersichtlich, dass diese Definition nicht sehr spezifisch ist; die Autonomität gilt hier als wesentliches Kriterium eines Agenten. Im Rahmen des IBM Agent Projects wird eine Definition verwendet, die den Anwender als Repräsentanten eines Agenten als zusätzliches Kriterium aufführt:

> *"Intelligente Agenten sind Softwareeinheiten, die eine bestimmte Menge von Operationen im Auftrag eines Anwenders bzw. anderen Programms mit einem bestimmten Grad an Unabhängigkeit bzw. Autonomie ausführen. Während sie dies tun, wenden sie Wissen des Anwenders an und repräsentieren seine Ziele und Wünsche. " (IBM, IBM Agent Project)*

Bei der Klassifikation von existierenden Softwareagenten wird deutlich, dass diese Definition nicht allen Konzepten gerecht wird.

Eine speziellere Definition vertritt Shoham in [Shoham, 1993]:

> *„ An agent is any entity whose state is viewed as consisting of mental components (e.g., beliefs, capabilities, choices, and commitments). “ (Shoham)*

Für ihn sind mentale Eigenschaften ausschlaggebendes Kriterium für Softwareagenten.

2.2.2. Eigenschaften von Softwareagenten

Als weitgehend akzeptiert gilt die in [Wooldridge, Jennings, 1995a] vorgestellte, so genannte schwache Charakterisierung (in [Shoham, 1993] wird eine strengere Charakterisierung vorgestellt), an der ein Softwareagent gemessen werden kann:

- **Autonomie:** Softwareagenten sind in der Lage, mit einem gewissen Maß an Eigenständigkeit einer Aufgabe nachzugehen.
- **Soziale Fähigkeiten:** Agenten interagieren mit anderen Agenten oder dem Benutzer.
- **Reaktivität:** Agenten nehmen ihre Umwelt wahr und reagieren rechtzeitig und angemessen auf Änderungen in ihrer Umwelt.
- **Proaktivität:** Agenten sind nicht nur in der Lage zu reagieren, sondern auch zu agieren: auf Änderungen in ihrer Umwelt können sie zielgerichtet die Initiative zu ergreifen.

2.2.2.1. Autonomie

Software benötigt in der Regel eine direkte Intervention des Anwenders, um Aufgaben auszuführen. Die Autonomie eines Softwareagenten erlaubt es ihm, eine gewisse Eigenständigkeit zu zeigen. Diese erfordert in den meisten Fällen, dass dem Agenten ein Modell seiner Umgebung vorliegt, sowie die Fähigkeit, auf Änderungen der Umwelt selbständig zu reagieren. Tatsächlich ist das **Bewusstsein** des Agenten (das Wissen über die eigene Rolle innerhalb der Umwelt) jedoch nicht zwingend notwendig um eine Autonomie zu erreichen. Die autonome Verhaltensweise gilt als wichtigstes Kriterium eines Softwareagenten.

2.2.2.2. Soziale Fähigkeiten (social ability)

Mit den sozialen Fähigkeiten von Agenten wird das Ziel verfolgt, mit mehreren kollaborierenden Softwareagenten eine höhere Leistungsfähigkeit zu erreichen, als mit einzelnen Agenten. Diese Hypothese wurde bereits als Ziel in der Verteilten Künstlichen Intelligenz formuliert [Huhns, Singh, 1994]:

$$\sum_i(L(agent_i)) > max(L(agent_i))$$

Abbildung 2: Leistungsfähigkeit kollaborativer Softwareagenten

Wobei L die Leistungsfähigkeit bezeichnet und sich unter anderem auf Attribute wie Geschwindigkeit, Zuverlässigkeit, Anpassungsfähigkeit, Korrektheit, Verhalten im Grenzbereich oder Kombinationen aus diesen beziehen kann. Die Funktion max() steht für die Ermittlung eines Maximums.

Gründe für den Einsatz kollaborierender Softwareagenten (nach [Nwana, 1996]):

- Die Lösung von Problemen, für die zentralisierte, einzelne Agenten aufgrund beschränkter Ressourcen oder dem Risiko der Zentralisierung nicht geeignet sind.

- Die Möglichkeit, bereits existierende Systeme (z.B. Expertensysteme) auf der Basis von Agenten zu vernetzen.
- Im Fall von räumlichen Trennungen von Systemen, z.B. Flugverkehrkontrollen oder verteilte Messstationen.
- Für Anwendungen, die auf verteilten Informationsquellen aufbauen, beispielsweise verteilte Online-Informationsquellen.
- Aus Gründen der Modularität (und somit einer verringerten Komplexität), der Verarbeitungsgeschwindigkeit (durch Parallelisierung), der Robustheit (durch Redundanz), der Flexibilität (neue Aufgaben können in modular organisierten System einfacher übernommen werden) und der Wiederverwendbarkeit von Wissen (durch gemeinsame Wissens-Ressourcen).

Die Kommunikation zwischen Agenten geschieht mittels einer Art Agentenkommunikationssprache.

Kommunikation zwischen Agenten

Eine Agentenkommunikationssprache (Agent Communication Language (ACL)) soll sowohl dem Austausch von Wissen zwischen Agenten dienen als auch für eine Wissensrepräsentation innerhalb einzelner Agenten taugen.

Eine Sprache zur Kommunikation zwischen Agenten muss zum Teil recht komplexen Anforderungen gewachsen sein (nach [Finin, Labrou, 1995]):

- **Form:** deklarativ, syntaktisch einfach und von Menschen lesbar
- **Inhalte:** Aufbau in Schichten – insbesondere Trennung von Sprachbestandteilen, die der Kommunikation dienen und solchen, die sich auf Kommunikationsinhalte beziehen

- **Semantik:** Die Übermittlung von Inhalten soll sich an Sprachtheorien der natürlichen Kommunikation orientieren. Eine Mehrdeutigkeit von Formulierungen soll ausgeschlossen sein.
- **Implementation:** Aufgrund der teilweise begrenzten Rechenleistung und Bandbreiten in potentiellen Einsatzgebieten (PDAs, Handys) soll eine Implementation effektiv sein. Partielle Implementationen bestimmter Sprachbestandteile sollen möglich sein.
- **Netzwerktauglichkeit:** Eine Agentenkommunikationssprache soll mit modernen Netzwerkkonzepten zu vereinbaren sein. Dazu gehören beispielsweise PPP, multi-/broadcasting, synchrone/asynchrone Verbindungen sowie die Unabhängigkeit vom Transportmechanismus.
- **Umgebung:** Es liegt eine verteilte, heterogene und dynamische Umgebung vor, in der eine Agentenkommunikationssprache zum Einsatz kommen kann.
- **Robustheit:** Eine sichere und verschlüsselte Kommunikation soll möglich sein.

Sprachen im Allgemeinen, d.h. auch natürliche Sprachen, ergeben sich aus Syntax (Grammatik – Regeln zur Form, Anordnung der Token etc.), Semantik (Ontologie – Bedeutung und Auslegung der Token) und Pragmatik (Anwendung). Neben den syntaktischen Regeln, die den kommunizierenden Agenten bekannt sein müssen, muss eine einheitliche Ontologie herrschen. Auch die Ontologie wird mittels der Agentenkommunikationssprache beschrieben. Die Kommunikation zwischen Agenten wird in mehrere Schichten gegliedert:

- **Transportschicht:** der Austausch von Nachrichten kann über unterschiedliche Konzepte erfolgen, die Wahl eines konkreten

Transportweges hängt von den Anforderungen der entsprechenden Agenten ab. Eine gängige Methode ist die Übermittlung via SMTP (per E-Mail).

- **Protokollebene:** Agenten können über standardisierte Protokolle wie die Knowledge Query and Manipulation Language (KQML, [Finin, 1991]) kommunizieren. Auf dieser Ebene wird bereits von der Transportschicht abstrahiert, d.h. es spielt keine Rolle, auf welchem Weg die Daten tatsächlich übermittelt werden. In Beispiel 1 wird der Austausch von Informationen innerhalb der Protokollebene verdeutlicht. Es werden Metadaten zur Nachricht ausgetauscht: Identität von Sender und Empfänger, Verweis auf die verwendete Sprache zur Wissensrepräsentation (z.B. Prolog), verwendete Ontologie (z.B. Thema: „Veranstaltungen in Leipzig“). Auch die eigentliche Nachricht wird innerhalb der Protokollebene übermittelt, jedoch nur syntaktisch. Eine semantische Erschließbarkeit der Nachricht erfordert weitere Ebenen.

- **Inhaltliche Ebene:** auf dieser Ebene werden Informationen darüber ausgetauscht, wie das Wissen, d.h. die Semantik der verwendeten Token, repräsentiert werden soll. Man kann dies mit dem **Kontext** in natürlichen Sprachen vergleichen: redet ein Erzähler von einem „Jaguar auf der Autobahn“, so wird dies vom Zuhörer anders interpretiert, als die Erzählung von einem „Jaguar im Zoo“. Damit auch Agenten „nicht aneinander vorbeireden“, wird die Semantik der Sprache während der Kommunikation eindeutig festgelegt. Eine in diesem Zusammenhang oft benutzte Sprache zur Übermittlung von Ontologien ist das Knowledge Interchange Format (KIF), entwickelt an der Standford University. KIF basiert auf Formulierungen der Prädikatenlogik 1. Stufe und ermöglicht den Wissensaustausch zwischen System der Künstlichen Intelligenz. Im

Beispiel 2 wird eine KQML Nachricht gezeigt, die für die Inhaltsangabe KIF einsetzt.

```
(KQML-performative

:sender   <word>
:receiver <word>
:language <word>
:ontology <word>
:content  <expression>
...)
```

Beispiel 1: Grundstruktur einer KQML-Nachricht

```
(tell

:sender john
:receiver lisa
:language KIF
:ontology family
:content ( <= (grandparent ?x ?z) (and (parent ?x ?y)
(parent ?y ?z)) )
)
```

Beispiel 2: Beispiel einer KQML-Nachricht unter Verwendung von KIF als Wissensrepräsentationssprache

Die sozialen Fähigkeiten sind keine notwendigen Eigenschaften für Agenten, da sie für die Autonomie nicht erforderlich sind. So kann ein Softwareagent auch isoliert seinen Zielen nachgehen und dabei intelligentes Verhalten zeigen.

2.2.2.3. Reaktivität

Als weitere grundlegende Eigenschaft von Agenten gilt die Reaktivität. Darunter wird die Fähigkeit eines Agenten verstanden, auf Ereignisse in seiner Umgebung selbständig und angemessen zu reagieren. Dies erfordert die Fähigkeit, Ereignisse wahrnehmen zu können. Der Agent muss also über entsprechende Eingabesysteme verfügen (z.B. Sensoren). Eine **angemessene Reaktion** des Agenten auf eine Änderung in seiner Umgebung setzt außerdem

voraus, dass dem Agenten ein Modell der Umgebung vorliegt, in dem seine Beziehung zur Umwelt definiert wird. Für dieses Eigenschaft wurde in dieser Arbeit bereits eine Bezeichnung eingeführt: das Bewusstsein des Agenten. Da in der Literatur auch hier z.T. stark voneinander abweichende Standpunkte vertreten werden, soll darauf hingewiesen werden, dass man vom Bewusstsein nicht als ein für Agenten notwendiges Kriterium sprechen kann. Tatsächlich wurde bereits „intelligentes“ Verhalten von Agenten realisiert, ohne die Agenten mit einem Bewusstsein auszustatten [Brooks, 1991b]. Unabhängig von der Implementation der Reaktivität kann man jedoch feststellen: ein System ohne die Fähigkeit zu reagieren ist nicht autonom, da sein Verhalten nicht durch ihn selbst mitbestimmt wird.

Einige Wissenschaftler vertreten die Auffassung, dass die Reaktivität über eine reine Reflexhandlung hinausgehen muss. Ein implizites Ansprechverhalten [Laurel, 1997] auf der Grundlage von Abwägungsprozessen anhand des/der Ziele(s) des Agenten ermöglicht eine solche fortgeschrittenere Form der Reaktivität.

2.2.2.4. Proaktivität

Neben der Eigenschaft, auf Veränderungen reagieren zu können, zeichnen sich intelligente Wesen dadurch aus, auch im Vorfeld reagieren zu können: zu agieren – selbständig die Initiative zu ergreifen. Man erwartet diese Fähigkeit auch von einem Softwaresystem, das als Softwareagent klassifiziert werden soll. Hier verhält es sich ähnlich wie mit der Reaktivität: der Agent muss entweder ein Bewusstsein haben oder so auftreten als hätte er ein Bewusstsein. Des Weiteren ist ein Ziel notwendig, das der Agent mit seinen Handlungen verfolgt, damit er auf Basis der Zustände und der Zielfunktionen Entscheidungen treffen kann.

2.2.2.5. Mobilität

Unter Mobilität (oder: Beweglichkeit) wird die Fähigkeit eines Agenten verstanden, seine Ausführungsumgebung selbständig zu ändern.

In populärer Literatur zum Thema Softwareagenten wird das Bild vermittelt, ein Schlüsselkonzept der Agententechnologien wäre die Beweglichkeit der Agenten. Tatsächlich ist die Beweglichkeit, wie auch die sozialen Fähigkeiten, für einen Softwareagenten nicht notwendig. Faktisch verzichtet man oft auf die Implementierung der Mobilität, da damit keine qualitative Verbesserung erreicht werden kann oder der Aufwand und die Risiken, die damit zusammenhängen, keinen Einsatz rechtfertigen. Es gibt jedoch Anwendungen, bei der aus praktischen Gründen (z.B. Bandbreite) die Mobilität wichtig ist. Bei der Beschreibung von Beispielanwendungen soll näher auf die Mobilität eingegangen werden.

2.2.3. Klassifikation von Softwareagenten

Eine Einteilung von Softwareagenten kann nach unterschiedlichen Kriterien vorgenommen werden. Besondere Aufmerksamkeit soll der Klassifikation nach [Nwana, 1996] gewidmet werden. In dieser Arbeit wurde die Evidenz der Softwareagenten als Klassifikationskriterium gewählt, d.h. das Vorkommen in der Literatur und der Wirtschaft.

Natürlich gibt es weitere Möglichkeiten der Klassifikation von Softwareagenten, u. a.:

- **Nach Aufgabe:** z.B. search agents, report agents, presentation agents, navigation agents, role-playing agents, management agents, search and retrieval agents, domain-specific agents, development agents, analysis and design agents, testing agents, packaging agents and help agents [King, 1995]

- **Nach Art der Aktivität:** reaktiv / proaktiv

- **Nach Mobilität:** statisch / mobil

Da einzelne Konzepte der Agenten wie die Mobilität, die Aktivität oder die sozialen Fähigkeiten in vielen existierenden Agentensystemen nur schwierig zu trennen sind, ist eine Einteilung nach der Evidenz an dieser Stelle vorteilhaft.

In Abbildung 3 werden die Agenten nach Eigenschaften dargestellt, die ein gewisses intelligentes Verhalten ermöglichen. Eine Interpretation dieser Abbildung kann wie folgt geschehen: Hybride- / Heterogene Agenten können über höhere soziale Fähigkeiten verfügen und autonomeres Verhalten aufweisen als alle anderen dargestellten Agentenkonzepte. Reaktive Agenten sind nur mit beschränkten sozialen und autonomen Fähigkeiten ausgestattet etc. Es sei darauf hingewiesen, dass die Abbildung nur qualitativer Natur ist. Es ist nicht auszuschließen, dass einzelne existierende Agenten damit nicht zu vereinbaren sind.

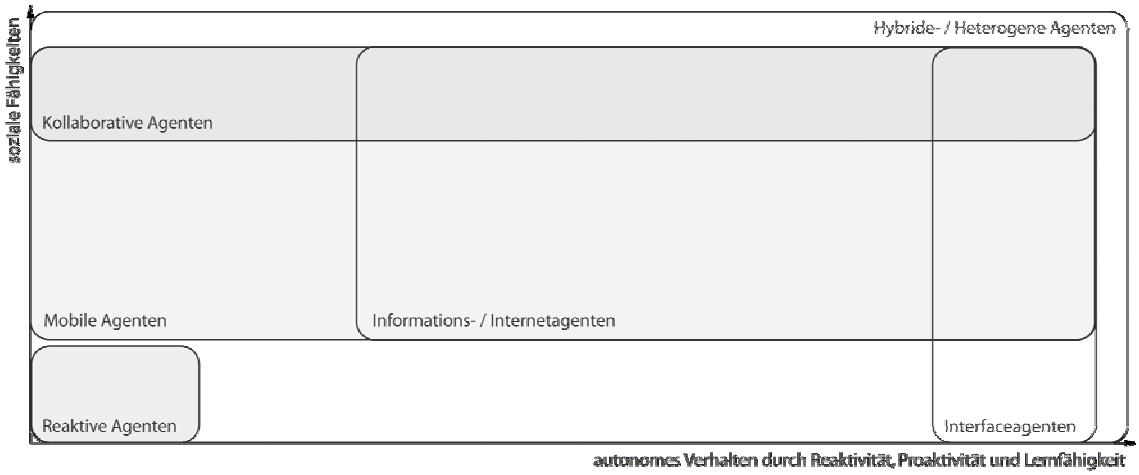

Abbildung 3: Agenten nach autonomen und sozialen Fähigkeiten

Kollaborative Agenten

Das Hauptmerkmal dieser Klasse von Agenten sind die komplexen sozialen Fähigkeiten, die eine Kommunikation unter den Agenten ermöglicht. Die Kommunikation beschränkt sich dabei nicht nur auf den Austausch von Informationen, sondern schließt fortgeschrittene Techniken wie etwa

Verhandlungen zwischen Agenten ein. Da auch Agenten, die man im Rahmen dieser Klassifikation eher anderen Klassen zuordnen würde, kollaborativ auftreten können (z.B. miteinander kommunizierende Informationsagenten), sind diese fortgeschrittenen sozialen Fähigkeiten eine wichtige Eigenschaft für kollaborative Agenten.

Beispiel: eine intelligente, agentenbasierte Architektur für das Management heterogener Netze [Gürer, Lakshminarayan, Sastry, 1998]

Das Management heutiger heterogener Netze stellt eine große Herausforderung für Tools des Netzwerkmanagement dar. Einzelne Managementkonzepte können den vielschichtigen Anforderungen nicht gerecht werden, eine Kombination verschiedener Konzepte ist daher notwendig. In dieser Arbeit wird ein Rahmenkonzept vorgestellt, das diese Probleme unter Verwendung von Softwareagenten angeht. Da diese Agenten starke soziale Fähigkeiten aufweisen, soll diese Arbeit als Beispiel für eine kollaborative Agententechnik dienen.

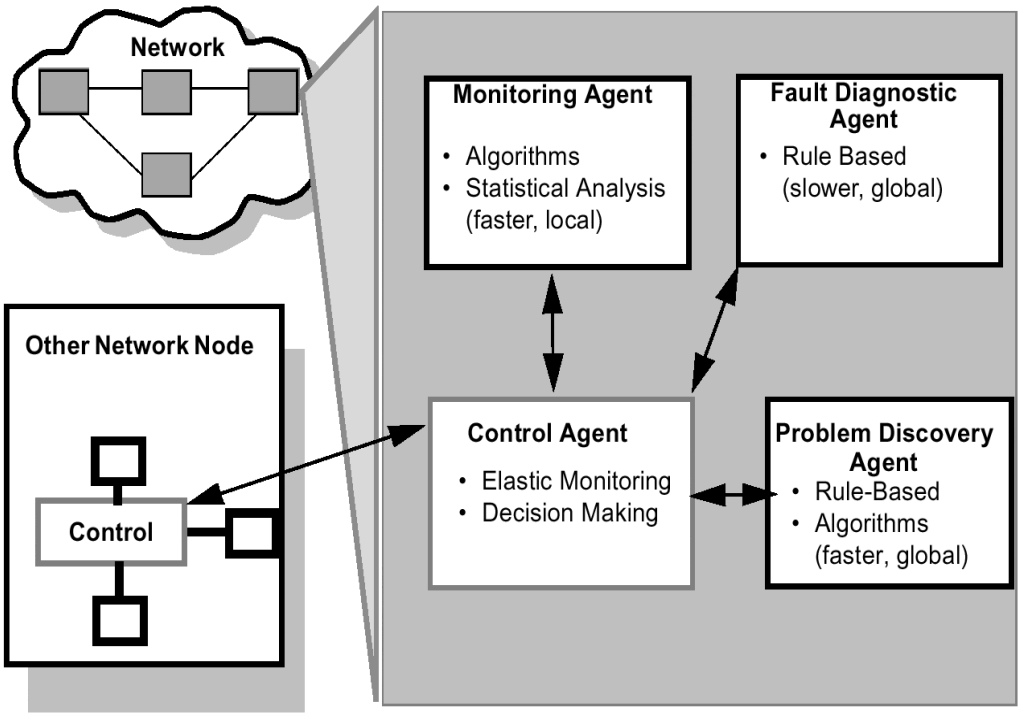

Abbildung 4: agentenbasierte Architektur eines Netzwerkmanagementsystems [Gürer, Lakshminarayan, Sastry, 1998]

Der Control Agent stellt die Zusammenarbeit der Agenten unter der Zielsetzung optimaler Quality of Services (QoS) des Netzwerkes sicher. Der Fault Diagnostic Agent ist für die Erkennung von langfristigen Fehlfunktionen im Netzwerk zuständig. Der Monitoring Agent beobachtet die Leistungsfähigkeit des Netzwerk. Vom Problem Discovery Agent werden kurzfristige Fehlfunktionen erkannt.

Die bemerkenswert kollaborative Eigenschaft des Multiagentensystems beruht vor allem auf der Fähigkeit der Control Agents, sowohl die Informationen der lokalen Agenten als auch Informationen von Control Agents anderer Knoten in den Entscheidungsprozess einbeziehen zu können.

Interfaceagenten

Benutzer des Microsoft Office-Paketes sind mit den so genannten Office Assistenten, die in Form von Büroklammern oder Hauskatzen auf sich aufmerksam machen, mehr oder weniger vertraut. Wenngleich sich der Nutzwert dieser konkreten Assistenten zumindest dem fortgeschrittenem Anwender nicht sofort erschließt, so kann man sich anhand dieser Beispiele eine gute Vorstellung über die Idee der Interfaceagenten machen.

Die Hauptaufgabe von Interfaceagenten ist typischerweise die Unterstützung eines Nutzers bei der Handhabung von Applikationen oder Betriebssystemen bzw. der Einarbeitung in solche Systeme. Dazu wird das Verhalten des Nutzers beobachtet, um ihn gegebenenfalls auf optimalere Wege aufmerksam zu machen, das System zu bedienen. Der Agent kooperiert mit dem Nutzer als autonomer, persönlicher Assistent und unterstützt ihn bei der Lösung konkreter Anwendungsaufgaben. Ziel ist es, dem Nutzer bestimmte Arbeitsschritte abzunehmen, um ungeübten Anwendern auch komplexe

Funktionalitäten zugänglich zu machen oder fortgeschrittenen Nutzern vorhersehbare Arbeitsschritte abzunehmen. Die Hypothese besteht darin, dass der Agent diese delegierten Aufgaben so erfüllt, wie der Anwender es von ihm erwartet.

Foner prägte die folgende Definition für Interfaceagenten: *"as opposed to any kind of agent, [an interface agent] is one that uses machine-learning techniques to present a pseudo 'intelligent' user interface for its actions" [Foner, 1993].* Wie auch in anderen Definitionen wird deutlich, dass man von Agenten ein gewisses intelligentes Verhalten erwartet, das beispielsweise fortgeschrittene Konzepte der KI, wie das des Maschinellen Lernens, erfordert.

Beispiel: Aria - ein Agent zum Verschlagworten und Durchsuchen von Bildern [Liebermann et al., 2001]

Mit der zunehmenden Verbreitung von Digitalkameras spielen digitale Bilder eine immer größere Rolle. Einige Anwender nutzen Bilder gelegentlich, um diese zu (meist privaten) E-Mails hinzuzufügen, d.h. Texte mit passenden Bildern zu versehen. Auch im professionellen Bereich gibt es einen Bedarf am Illustrieren von Texten. Dies erfordert in der Regel einen relativ hohen manuellen Aufwand: Bilder müssen archiviert, annotiert (verschlagwortet) und schließlich recherchiert werden. Aria wurde konzipiert, um den Anwender bei diesen Aufgaben zu unterstützen.

Während der Anwender beispielsweise eine E-Mail formuliert, analysiert Aria die textuellen Eingaben. Der Agent versucht zu den letzten n Wörtern passende Bilder in einer ihm bekannten Bilddatenbank zu finden. Wird er fündig, so werden die Bilder in einem separaten Fenster angezeigt (sortiert nach Relevanz) und können vom Anwender per drag & drop in die E-Mail eingefügt werden – ohne dass er dazu den Fokus von der E-Mail nehmen muss, wie es bisher der Fall war.

Die Lernfähigkeit des Agenten und somit das intelligente Verhalten wird durch die Fähigkeit erreicht, Annotationen selbständig zur Bilddatenbank hinzuzufügen. Nehmen wir an, ein Nutzer möchte ein Bild einfügen, welches nicht bereits vom Agenten vorgeschlagen wurde. In diesem Fall wählt er eigenhändig das Bild. Der Agent untersucht nun den textuellen Kontext des Bildes, d.h. die vom Nutzer soeben zuvor formulierten Wörter, und registriert diese als mit dem Bild verwandt.

Diese Wiederverwendung von Nutzereingaben ist ein wichtiger Aspekt in Agentenkonzepten im Allgemeinen und eine entscheidende Methode zur Reduzierung des Nutzeraufwandes. [Liebermann et al., 2001]

Mobile Agenten

Alle bisher besprochenen Agenten waren nicht notwendigerweise Mobile Agenten. *Nicht notwendigerweise* deshalb, weil die entscheidenden Funktionalitäten nicht von der Mobilität des Agenten abhingen. An einem speziellen Anwendungsfall soll das Konzept der mobilen Agenten verdeutlicht werden.

Gegeben sei ein Agent, der einen Anwender bei der Informationssuche im Internet unterstützt. Der Anwender teilt dem Agenten bestimmte Schlüsselwörter mit, anhand derer die Informationen abgefragt werden sollen. Der Agent greift nun über ein Netzwerk auf die verteilten Informationsquellen zu. Der Agent besitzt die Fähigkeit, selbständig Entscheidungen zu treffen und seine Ergebnisse anhand des Feedbacks des Anwenders zu verbessern, kurz: er ist autonom und lernfähig – Eigenschaften, die ihn als Agenten ausweisen.

Dieses Szenario ist nicht ungewöhnlich und realisierbar mit statischen Agenten. Was aber, wenn es sich bei den zu durchsuchenden Informationen z.B. um komplexes Bildmaterial handelt? Bei entsprechend großen Datenmengen (multimedialen Daten, Messdaten etc.) ist ein Transport dieser

Daten von der Datenquelle zum Agenten nicht mehr sinnvoll. In diesem Fall wäre es wünschenswert, dass der Agent seine Ausführungsumgebung während der Laufzeit selbständig ändern kann. Der Agent würde die Datenquellen lokal abfragen und zu seinem Ausgangspunkt zurückkehren.

Ein weiteres Szenario wären beschränkte Ressourcen in der Startumgebung des Agenten: etwa auf einem PDA oder einem Handy. Insbesondere bei zukünftigen Anwendungen ist es denkbar, dass ein Agent für ressourcenhungrige Aufgaben auf eine leistungsfähigere Umgebung ausweicht oder komplexe Aufgaben dynamisch auf verschiedene Rechner verteilt werden.

Bevor solche Szenarien jedoch Realität werden, sind noch viele grundlegende Fragen zu klären und Standards zu etablieren:

- Wie bewegt sich ein Agent von Ort zu Ort (inklusive Objekte, was geschieht mit Referenzen und Filehandles etc.)?
- Wie authentifiziert sich ein Agent und wie wird sichergestellt, dass ein Agent das ist, wofür er sich ausgibt?
- Wie können Mobile Agenten vor Manipulation oder Spionage geschützt werden?
- Wie kann die Funktionalität eines Agenten sichergestellt werden, d.h. wer garantiert, dass der Agent das macht, was er machen soll?

Beispiel: AJANTA – Forschungsprojekt zu Mobilen Agenten an der Universität von Minnesota [Tripathi, 2002]

AJANTA ist ein auf Mobilen Agenten basierendes System zur Überwachung von Netzwerken. Das Design des Systems wurde maßgeblich bestimmt durch den Wunsch nach einem Netzwerküberwachungskonzept das folgenden Anforderungen gerecht wird:

- Dynamische Erweiterbarkeit und Konfigurierbarkeit: das System sollte den dynamischen Eigenschaften eines Netzwerks (Änderungen der Hardware, Software, Nutzungsbedingungen) gerecht werden.
- Aktive Überwachung: die Überwachungseigenschaften sollten sich dem aktuellen Gegebenheiten anpassen. In kritischen Situationen sind ggf. intensivere Überwachungen notwendig.
- Dezentralisierte Datengewinnung: aus Gründen der Skalierbarkeit sollte es möglich sein, Überwachungsdaten dezentral zu sammeln, d.h. nicht von einem zentralen Rechner aus auf andere Ressourcen überwachend zuzugreifen, sondern an beliebigen Punkten des Netzwerkes diese Überwachung vornehmen zu können. Dies ist auch vorteilhaft bei der Verwendung mehrerer administrativer Domänen.
- Policy-basierte Überwachung: die Überwachungsziele sollten über High Level Policies definiert werden können. So sollte beispielsweise spezifizierbar sein, dass kein Nutzer auf mehr als zwei Accounts zugreifen darf.
- Sichere Überwachungsumgebung: das Überwachungssystem selbst soll vor Manipulationen geschützt sein.
- Performance: das Überwachungssystem sollte sich nicht störend auf die Leistungsfähigkeit der überwachten Hosts auswirken.

Um diese Funktionalitäten zu erreichen, wurde eine Java-basierte Architektur für die Implementation des Agenten gewählt. Code und Ausführungskontext werden gemeinsam mit den Daten des Agenten gekapselt, so dass eine Änderung des Ausführungsortes des Agenten stattfinden kann. Ein so

genannter Agentenserver ist an jedem Knoten des Netzwerkes notwendig, an dem ein Mobiler Agent ausgeführt werden soll.

Für einige der genannten Funktionalitäten wie die dezentrale Datengewinnung und die Performance-Anforderungen sind Mobile Agenten besonders geeignet. Für detaillierte Informationen soll an dieser Stelle auf [Tripathi, 2002] und [Appleby, 1994] verwieden werden.

Informations-/Internetagenten

Dass es trotz der in der Einleitung dieser Arbeit beschriebenen Situation bisher keine bedeutenden Fortschritte bei den Suchwerkzeugen gegeben hat (die meistgenutzten Suchwerkzeuge sind noch immer Suchmaschinen auf Volltextsuche-Basis), weist auf die hohen Anforderungen bei der Entwicklung solcher Werkzeuge hin. Informations- bzw. Internetagenten bieten viel versprechende Konzepte zur Bewältigung dieser Anforderungen.

> *"In the future, it [agents] is going to be the only way to search the Internet, because no matter how much better the Internet may be organised, it can't keep pace with the growth in information..." [Bob Johnson, analyst at Dataquest Inc.]*

Im Kapitel 2.2.3 wurden bereits Konzepte vorgestellt, die sich auch zur Realisierung fortschrittlicher Suchwerkzeuge in Informationsnetzen eignen:

- Lernfähigkeit: Art und Häufigkeit der Nutzung von Suchwerkzeugen sind geradezu prädestiniert dazu, Techniken des Maschinellen Lernens einzusetzen, um die Qualität des Suchwerkzeugs selbständig zu verbessern. So fällt ein Nutzer immer wieder Entscheidungen über die Qualität der von der Suchmaschine ausgegebenen Suchergebnisliste, indem er auf einen Link innerhalb dieser Liste klickt. Herkömmliche

Suchmaschinen nutzen diese Informationen kaum. Eine Suche nach einem Begriff des aktuellen Zeitgeschehens über die Suchmaschine Google bringt neben gefundenen Webseiten oft auch Ergebnisse in speziell durchsuchten Nachrichten-Informationsquellen. Die Links zu diesen Suchergebnissen führen gelegentlich zunächst auf den Google-Server, wodurch es möglich wird, das Feedback des Anwenders auf eine Suchanfrage auszuwerten. Im einfachsten Fall könnte man schlussfolgern, dass das Ranking der Suchergebnisse mangelhaft ist, wenn die meisten Nutzer von dem an erster Stelle auf der Suchergebnisseite aufgeführten Link keinen Gebrauch machen. Beim Durchsuchen von Webseiten werden die Interaktionen der Anwender allerdings nicht ausgewertet; die Links zu den Suchergebnissen führen direkt zu den Zielservern. Klicks auf diese Links sind damit für die Google-Technologie nicht nachvollziehbar. Gründe für den Verzicht auf diese Informationen können im erhöhten Leistungsbedarf oder in der weit verbreiteten Skepsis gegenüber Techniken der Auswertung persönlicher Daten liegen.

- Flexibilität: Aufgrund der Veränderlichkeit der Informationsquellen sind gewisse adaptive Fähigkeiten notwendig, die es dem Agenten erlauben, sich Änderungen z.B. in der Struktur der Informationsdarstellung anzupassen. In Agentenkonzepten wird eine solche Flexibilität über reaktive und proaktive Eigenschaften erreicht, die es den Agenten erlauben, sich seiner Umgebung anzupassen.

- soziale Fähigkeiten: Informationssuche ist eine sehr individuelle Tätigkeit, bei der die persönlichen Präferenzen des Informationssuchenden eine zentrale Rolle spielen. Diese Präferenzen lassen sich nicht mittels einiger Schlüsselwörter ausdrücken, wie es

aktuelle Suchmaschinen von den Anwendern erwarten. Die sozialen Fähigkeiten eines Agenten statten ihn mit der Fähigkeit aus, mit seinem Anwender auf einer höheren Ebene zu kommunizieren als per unidirektionaler Schlüsselwortübermittlung. Auch eine Kommunikation unter Agenten, wie es bei dem Konzept der Kollaborativen Agenten beschrieben wurde, kann zur Qualitätssteigerung des Suchwerkzeuges beitragen.

- Mobilität: Spezielle Informationsquellen wie datenreiche Bilddatenbanken können es erforderlich machen, dass ein Suchwerkzeug „vor Ort“ sucht, d.h. lokal in der Umgebung der entsprechenden Datenbank ausgeführt wird. Mobile Agenten sind dazu in der Lage, wenn die Zielumgebungen darauf vorbereitet sind.

Beispiel: ARIADNE - web-based information integration [Knoblock, 2001]

Die meisten Suchwerkzeuge wurden für eine spezielle Informationsquelle entwickelt, etwa allgemeine Teile des WWW, spezielle Teile des WWW (z.B. nur Nachrichtenseiten), das Usenet oder spezielle Datenbanken. Möchte nun ein Anwender nach Informationen suchen, so muss er sich zunächst für ein Suchwerkzeug und somit für eine Quelle entscheiden. Eine zentrale Idee des ARIADNE Konzeptes ist es, zwischen dem Anwender und den Informationsquellen eine Abstraktionsschicht einzuführen: den so genannten Mediator (Vermittler). Der Mediator kapselt Eigenschaften der Informationsquellen wie Protokoll, Struktur oder Ontologie und bildet diese in einem einheitlichen Modell ab, für das der Anwender beispielsweise Suchanfragen formulieren kann.

Angenommen, es soll ein Web-basiertes Suchwerkzeug konstruieren werden, welches auf verschiedene Quellen zugreift, um als Restaurantführer zu dienen. Mögliche Informationsquellen wären verschiedene Seiten mit

Restaurantbewertungen, Seiten der Restaurants selbst und Stadtplandienste. Die ARIADNE-Technologie unterstützt den Entwickler eines solchen Suchwerkzeuges entscheidend bei der Integration dieser Quellen in eine gemeinsame Ontologie. Mit Hilfe dieser Ontologie kann nun nach Informationen gesucht werden, die zuvor auf verschiedene Quellen verteilt waren (z.B. „Zeige mir die Telefonnummer und den Weg zu einem Restaurant der Preisklasse A in Dresden-Neustadt!“).

Die Probleme, die sich der ARIADNE-Technologie stellen, sind vielfältig:

- Strukturierung von semistrukturierten Daten: Um Abfragen innerhalb der Daten gestalten zu können, müssen diese strukturiert werden. Webseiten sind in der Regel semistrukturiert. Ein Anlernen des Systems ist notwendig; in unserem Restaurant-Beispiel müsste der Entwickler des Suchwerkzeuges der Technologie beispielsweise vermitteln, wie die Telefonnummern der Restaurants auf den Seiten erscheinen. Dazu werden jedoch Lernmethoden eingesetzt, die der Technologie eine gewisse Eigenständigkeit verleihen – eine Eigenschaft, die Softwareagenten kennzeichnet.

- Planungsprozess: Auch nach der Strukturierung der Daten können Abfragen nicht einfach so vorgenommen werden, wie es etwa bei relationalen Datenbanken der Fall ist. Die Abfrage „Zeige mir den Weg zu den Restaurants“ erfordert einige Zwischenschritte, die die Technologie selbständig absolvieren muss. Dies erfordert die Fähigkeit des Systems, selbständig Entscheidungen zu treffen.

- Optimierung: Um den Zugriff auf zum Teil zahlreiche Ressourcen zeitlich zu optimieren, werden bestimmte Daten lokal gespeichert. Die Auswahl dieser Daten hängt von der Update-Frequenz der entsprechenden Quellen und von den Abfragen des Nutzers ab. Das Up-

To-Date-Bleiben ist übrigens ein wichtiges Problem der Informationsagenten im Allgemeinen.

- Inkonsistenzen in der Namensgebung: In dem Restaurant-Beispielfall wäre es denkbar, dass eine Gaststätte auf einer Seite mit dem Namen „Art's Deli" und auf einer anderen Seite mit dem Namen „Art's Delicatessen" bezeichnet wird. Um solche Inkonsistenzen zu erkennen, werden von der ARIADNE-Technologie Mapping-Methoden eingesetzt, die unter anderem auf Verfahren des Statistischen Lernens basieren.

Das ARIADNE-Konzept kann bei der Entwicklung eines Information Agent sehr hilfreich sein. Dabei kommen Methoden zum Einsatz, die eine Klassifikation dieser Technologie als Softwareagent zulassen.

Reaktive Agenten

Bisher wurden Agentenkonzepte betrachtet, deren fortgeschrittenes Verhalten aus aufwendigen Konzepten resultierte. Die Nachteile gegenüber schlanken Agentenkonzepten liegen auf der Hand:

- hohe Entwicklungs- und Wartungskosten
- hoher Ressourcenbedarf
- geringe Flexibilität
- geringe Robustheit und Fehlertoleranz (z.B. bei Ausfall eines Agenten)

Die Komplexität der Agenten ist jedoch notwendig, um das gewünschte Verhalten zu erreichen. Dass ein gewisses intelligentes Verhalten jedoch auch auf der Basis relativ einfacher Agenten realisierbar ist, beweisen die so genannten Reaktiven Agenten. Die Hypothese des Konzeptes der Reaktiven Agenten ist, dass intelligentes Verhalten auch mit einfachen Agenten erreicht

werden kann, indem das Zusammenspiel dieser Agenten ausreichend gut organisiert ist. Proaktivität (das Ergreifen der Initiative), ausgeprägte soziale Fähigkeiten (Kommunikation zwischen Agent/Agent oder Agent/Mensch auf einem höheren Sprachlevel) oder Mobilität sind dazu nicht notwendig. Reaktive Agenten planen nicht voraus, ihre wesentlichen Eigenschaften sind die Reaktivität (das Reagieren auf ihre Umwelt) und die gute Organisation des Zusammenspiels der einzelnen Agenten. Natürliche Phänomene wie Fischschwärme oder Ameisenkolonien können mit Reaktiven Agenten simuliert werden.

Beispiel: MANTA Projekt – Modellierung und Simulation des sozialen Verhaltens einer Ameisenkolonie [Drogoul, 1995]

Dass komplexe Strategien aus einfachem taktischem Verhalten entstehen können, lässt sich anhand des Verhaltens einer Ameisenkolonie studieren. Im Projekt MANTA wurde jeder Organismus der Population (Ameise, Puppe, Larve und Ei) durch einen Agenten dargestellt, der alle Verhaltensweisen des Organismus widerspiegelte, die für die Simulation notwendigen waren. Um einen Aussage über die Qualität des Verhaltens treffen zu können, wurde dies mit dem natürlichen Verhalten der nachempfundenen Spezies (unter Laborbedingungen) verglichen.

Im Experiment gab es keine veränderlichen Umweltbedingungen, die einzige Variable bestand in den zufälligen Bewegungen der Ameisen. Alle Experimente wurden mit dem Platzieren einer Königin begonnen und beendet, wenn die Königin entweder verhungerte, oder wenn eine kritische Menge an herangezüchteten Ameisen überschritten wurde.

Die Quote des Überlebens einer virtuellen Kolonie lag in diesem Experiment erstaunlich nah bei der Quote der natürlichen Ebenbilder. Interessanterweise konnte bei der Beobachtung der Königin ein Verhalten festgestellt werden, das

über die Fähigkeiten hinausging, die dem entsprechenden Agenten tatsächlich mitgegeben wurden. Es bildete sich eine Langzeitstrategie aus dem Zusammenspiel des Verhaltens der Königin und den Bedürfnissen der Brut. Die Hypothese, komplexe Strategien könnten aus einfachem taktischen Verhalten entstehen, konnte in diesem Experiment bestätigt werden.

Hybride und Heterogene Agenten

Der Vollständigkeit halber sei auch erwähnt, dass es Agenten gibt, die auf der Kombination verschiedener Agentenkonzepte basieren. So kann ein Agent, dessen soziale Fähigkeiten ebenso ein Hauptmerkmal ist wie die Mobilität, als Hybrid bezeichnet werden. Heterogene Agentensysteme sind solche, die aus verschiedenen Agenten unterschiedlicher Klassen bestehen, beispielsweise ein System, welches aus dem Zusammenspiel kollaborativer Agenten und mobiler Agenten profitiert.

Die Hypothese beider Ideen ist die, dass das resultierende System über Eigenschaften verfügt, die mit einzelnen Agentenkonzepten nicht realisierbar sind.

2.2.4. Softwareagenten als WWW-Assistenten

2.2.4.1. Letizia [Liebermann, 1995]

Als Beispiel für einen WWW-Assistenten auf der Basis eines Softwareagenten soll auf Letizia hingewiesen werden, ein Forschungsprojekt des MIT Media Laboratory. Letizia ist ein Agent, der den Nutzer beim Surfen im Internet unterstützt, indem er das Surfverhalten des Nutzers in einem konventionellen Webbrowser beobachtet. Nach unserer Klassifikation wäre eine Zuordnung zu den Interfaceagenten möglich.

Die Annahme des Konzeptes besteht darin, dass die Kooperation zwischen Nutzer und Agenten die Informationssuche im Internet nach den Vorstellungen des Nutzers verbessern kann. Letizia geht dabei folgenden Aufgaben nach:

Letizias Aufgaben

- Links innerhalb der Seiten, die der Nutzer betrachtet, werden vom Agenten automatisch verfolgt.
- Die Reihenfolge der Linkverfolgung wird vom vermuteten Interesse des Nutzers an diesen Links abhängig gemacht: der Link, an dem er wahrscheinlich am meisten Interesse hat, wird zuerst untersucht. Dabei kommen simple Heuristiken zum Einsatz.
- Die verlinkten Seiten werden untersucht und mit dem persönlichen Interessenprofil des Nutzers verglichen.
- Auf Nachfrage spricht der Agent Empfehlungen über die verlinkten Seiten aus.

Somit unterscheidet sich das Verhalten des Agenten im Prinzip nicht vom menschlichen Surfverhalten. Die Fähigkeit des Nutzers, den Inhalt verlinkter Seiten vorherzusagen, geht weit über die Möglichkeiten eines Agenten hinaus. Menschliche Nutzer verwenden dazu ihre Erfahrungswerte über den Zusammenhang zwischen den Merkmalen des Links (Linktext, Kontext, Zieladresse) und den Inhalten der verlinkten Seiten. Trotzt dieses qualitativen Nachteils der Agenten haben diese einen entscheidenden Vorteil: sie sind deutlich schneller. So kann der Agent bereits während sich der Nutzer einen Eindruck von einer Seite verschafft, die Links dieser Seite untersuchen.

Persönliches Interessenprofil des Nutzers

Um dem Nutzer inhaltliche Empfehlungen auszusprechen, muss der Agent natürlich in der Lage sein, das Interessenprofil des Nutzers zu modellieren. Letizia geht dabei zwei Ansätzen nach: dem des Information Retrieval, bei dem der Suche nach Informationen durch den Nutzers eine besondere Bedeutung beigemessen wird sowie dem Ansatz des Information Filtering, bei dem aus einer Menge von Informationen irrelevante Informationen herausgefiltert werden. Zum besseren Verständnis dieser Ansätze sollen zwei Beispiele angeführt werden. Klickt der Nutzer auf einen Link, so kann man den Linktext als Informationssuche betrachten. Der Klick kann mit der Eingabe des Linktextes in eine Suchmaschine verglichen werden, also mit einer Suche nach bestimmten Informationen. Techniken des Information Retrieval können in diesem Fall zum Einsatz kommen, um den Linktext zu evaluieren und das Interessenprofil des Nutzers anzupassen. Konzepte des Information Filtering werden hingegen verwendet, um ganze Webseiten zu analysieren. Abhängig von der Verweildauer auf diesen Seiten bzw. von verschiedenen Aktionen des Nutzers wie das Setzen eines Lesezeichens, nimmt der gefilterte Inhalt dieser Seite ebenfalls Einfluss auf das Interessenprofil.

Ein wichtiger Aspekt bei der Verwaltung eines Interessenprofils ist das **Vergessen**. Letizia berücksichtigt die verstrichene Zeit seit dem Surfen innerhalb bestimmter Themengebiete (und damit daraus abgeleiteten Interessengebieten). Wie bei vielen Anwendungen der KI wurde auch dieses Feature dem menschlichen Verhalten nachempfunden, um mit dem Agenten möglichst menschliches Verhalten zu imitieren.

2.2.4.2. WebWatcher [Armstrong et al., 1995]

Ein weiterer WWW-Assistenten auf Agentenbasis, der an dieser Stelle vorgestellt werden soll, ist der WebWatcher, ein Projekt der Carnegie Mellon University in Kooperation mit der Universität Dortmund.

Die Zielsetzung ist der des Agenten Letizia sehr ähnlich. Ein wesentlicher Unterschied besteht in der konkreten Implementation: während Letizia auf Client Seite operiert, wird der WebWatcher auf Serverseite zwischen den Zielserver und den Klienten geschaltet – ähnlich einem Proxyserver. Da die Anfragen nun über den WebWatcher-Server laufen, kann dieser die Seiten manipulieren, um beispielsweise Hinweise auf die Relevanz eines Links für den aktuellen Nutzer einzufügen.

Bestimmung der Relevanz von Hyperlinks für den Nutzer

Auch dieser Agent untersucht die Links der aktuellen Seite des Nutzers automatisch, um die Relevanz der Links anhand des persönlichen Interessenprofils des Nutzers zu bestimmen. Ein Schwerpunkt des Konzeptes liegt in den Lernmethoden, die es dem Agenten erlauben sollen, seine Qualität zu verbessern. Für jeden Link wird eine Zielfunktion ausgeführt, die die Wahrscheinlichkeit ausdrückt, ob ein Nutzer diesen Link klicken wird oder nicht (gegeben der aktuellen Seite und des Interessenprofils des Nutzers):

```
LinkQuality : Page x Interest x Link → [0,1]
```

Abbildung 5: Wahrscheinlichkeit der Linkverfolgung durch einen Nutzer

Lernmethoden des WebWatcher

Es sollen hier zwei interessante Lernmethoden des WebWatcher herausgegriffen werden, die die Qualität der Zielfunktion verbessern:

- Lernen von vergangenen Touren: zu Beginn einer Tour, d.h. eines Surfvorgangs des Nutzers, gibt dieser dem Agenten Schlüsselwörter an,

die seine Interessen während dieser Tour widerspiegeln sollen. Zwischen diesen Schlüsselwörtern und den Hyperlinks, die der Nutzer im Laufe der Tour klickt, kann der Agent nun eine Beziehung herstellen.

- Lernen anhand der Hypertextstruktur: anstatt nur die Folgeseite des Hyperlinks zu betrachten, ist der Agent in der Lage, noch tiefer in den Hyperlink-Graphen vorzudringen. Angenommen, ein Nutzer mit dem formulierten Interesse „Nachrichten aus Leipzig" befindet sich auf einer Seite zum Thema „Olympia in Leipzig". Auf dieser Seite befindet sich Link 1: „Nachrichten um den Olympiapark" und Link 2: „Leipzig Aktuell". Die Seite hinter Link 1 enthält einige aktuelle Nachrichten zum Thema Olympia in Leipzig und wurde aufgrund des Linktextes von vielen Nutzern mit diesem Interesse angeklickt. Jedoch ist diese Seite eine Sackgasse, während Link 2 auf viele weitere Links zum Thema „Nachrichten aus Leipzig" verweist. Mittels der Lernmethode 1 würde der Agent nicht bemerken, dass Link 2 somit indirekt relevanter für den Nutzer ist, deshalb bezieht diese Lernmethode mehr als eine Verbindung im Hyperlink-Graphen mit ein.

Anhand einer empirischen Untersuchung wurde die Leistungsfähigkeit des Agenten analysiert [Armstrong et al., 1995], vorherzusagen, welcher Link von einem Nutzer zu einem bestimmten Zeitpunkt auf einer Website geklickt würde. Bei einem Vergleich wurde diese Aufgabe von menschlichen Experten übernommen, die mit der Testseite vertraut waren: während diese die Klicks mit einer Genauigkeit von 47,5 % vorhersagten, kam der Agent auf erstaunliche 42,9 %.

2.2.5. Herausforderungen an Softwareagenten

Obwohl bereits sehr fortgeschrittene Agentensysteme existieren, bleiben einige wichtige Fragen offen. Zu den Problemstellungen, die die Konzepte der Softwareagenten mit sich bringen, gehören:

- **Datenschutz:** Eine Stärke der Agenten kann die Fähigkeit sein, den Anwender individuell zu behandeln. Diese Eigenschaft setzt voraus, dass der Agent eine Vorstellung vom Individuum hat: ein persönliches Profil des Anwenders. Es muss sichergestellt werden, dass dieses Profil vertraulich behandelt wird. Dies ist insbesondere dann eine Herausforderung, wenn ein Mobiler Agent mit diesem Profil in eine fremde Ausführungsumgebung migriert.

- **Haftbarkeit:** Eine Grundeigenschaft von Agenten ist die Autonomität. Eigenständiges Handeln impliziert das selbständige Treffen von Entscheidungen. Dies ermöglicht es den Agenten, fortgeschrittenen Aufgaben nachzugehen, ohne dass eine ständige Intervention eines Menschen erforderlich ist. Was aber, wenn sich eine solche Entscheidung im Nachhinein als Verhängnisvoll herausstellt (z.B. der Kauf eines teuren Tickets durch einen Kauf-Agenten)? Kann sichergestellt werden, dass Agenten in allen Situationen die richtigen Entscheidungen treffen und wenn nicht, wer haftet für eine solche Entscheidung – Benutzer oder Hersteller des Agenten? Es müssen rechtlichen Rahmenbedingungen für solche Fälle geschaffen werden.

- **Umgangsformen:** Mobile Agenten stellen eine besonders hohe Herausforderungen an Sicherheitskonzepte dar. Systeme, die die Ausführung von Mobilen Agenten zulassen, müssen sichergehen können, dass diese sich korrekt identifizieren und die Autorität des Servers akzeptieren (z.B. nur zugelassene Ressourcen nutzen und nur

zulässigen Aufgaben nachgehen). Dies gilt auch für die Kommunikation unter Agenten: wie kann garantiert werden, dass sich der Kommunikationspartner den gewünschten Regeln entsprechend ehrlich (keine bewussten Falschaussagen), kommunikativ (Antworten auf Anfragen) und kooperativ (Unterstützung eines fremden Agenten, z.B. durch Informationen, die er nicht explizit angefordert hat) verhält?

2.2.6. Ausblick zu den Softwareagenten

Es soll an dieser Stelle auf die Notwendigkeit einer Betrachtung neuartiger Konzepte der Programmierung aufmerksam gemacht werden, die eng mit den Ideen der Softwareagenten zusammenhängt.

In der heutigen Softwareentwicklung herrscht eine Situation, in der zahlreiche Gegebenheiten auf konzeptionelle Unzulänglichkeit konventioneller Programmierparadigmen hinweisen, u. a.:

- steigende Anforderungen an die Komplexität von Softwareprodukten und somit immer höherer Entwicklungsaufwand
- immer größere Menge an Protokollen, Schnittstellen und Konzepten, die Softwareentwickler überblicken müssen
- wachsender Anteil der Softwarewartung an der Softwareentwicklung (siehe Abbildung 6); Während in den 70er Jahren der Anteil der Wartungskosten an den Entwicklungskosten bei knapp 40% lag, lag er in den 90er Jahren bereits bei 75%.

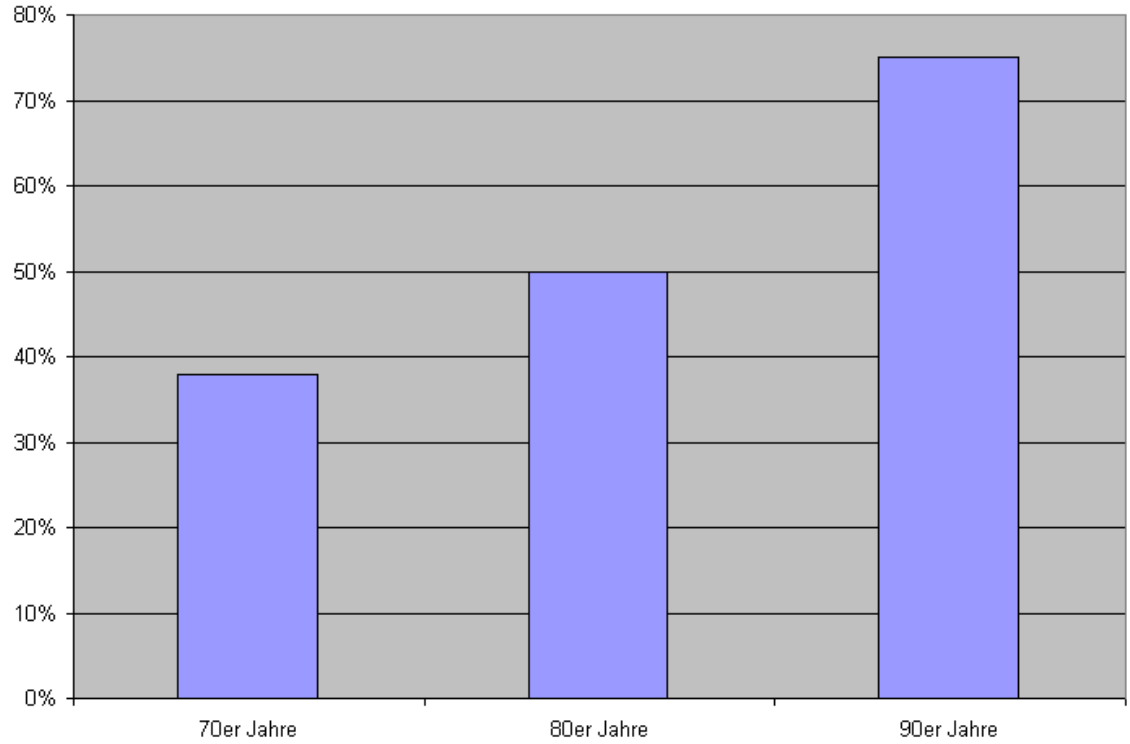

Abbildung 6: Anteil der Wartung an den Entwicklungskosten [Zuse, 1999a]

Während man in den Anfängen der modernen Computerentwicklung noch von „Programmierkunst“ sprach und das Erlernen der Syntax einer Programmiersprache im Vordergrund stand, wurde die Softwareentwicklung im Laufe der Jahre wissenschaftlich fundierter (Softwaredesign, Validierungsmöglichkeiten etc.). Auch die Entwicklungsumgebungen wurden zunehmend komfortabler (Developer Kits, Bibliotheken, Visuelle Komponenten etc.). Die Programmierung selbst jedoch läuft noch immer nach demselben Prinzip ab, wie es bereits vor 50 Jahren der Fall war: mittels Formulierung konkreter Anweisungen. Konkrete Anweisungen, die zwar keinen Spielraum für Interpretationen offen lassen (was im Allgemeinen auch erwünscht ist), die aber sehr detailliert ausfallen müssen, um umfangreiche Funktionalitäten zu erfüllen.

Seit den ersten Ideen der Programmierung (Charles Babbage, 1833) über die erste niedergeschriebene Programmiersprache "Plankalkül" (Konrad Zuse,

1941) bis zu heutigen so genannten Hochsprachen wurde stets in konkreten Anweisungen programmiert, d.h. Programmierer mussten sich auf der kleinsten funktionalen Ebene der jeweiligen Programmiersprache bewegen.

Es hat eine Abstraktion von der zugrunde liegenden Hardware gegeben. Aus Gründen mangelnder Leistungsfähigkeit der Rechner wurden Computersysteme bis in die 80er Jahre sehr maschinennah programmiert: mittels Programmiersprachen der 1. (Maschinensprachen) oder 2. (Assemblersprachen) Generation. Heute ist dies nur noch selten der Fall.

Die Idee der Logischen Programmierung, Problemstellungen aus einer deklarativen Sicht zu beschreiben (z.B. mittels Formulierung von Regeln und Fakten zu einer Problematik) im Gegensatz zur herkömmlichen prozeduralen Sichtweise (Abarbeiten aufeinander folgender Programmschritte) konnte sich nur in speziellen Anwendungsgebieten wie etwa in Expertensystemen oder zu Forschungszwecken durchsetzen.

Mit dem allmählichen Siegeszug des Paradigmas der objektorientierten Programmierung in kommerziellen Anwendungen seit der Einführung der ersten rein objektorientierten Sprache SMALLTALK durch Alan Kay wurde auch eine Abstraktion auf der Ebene der Daten möglich. Im Gegensatz zur Funktionellen Programmierung wurden nun nicht mehr die Funktionen, sondern die Objekte in den Mittelpunkt der Programmierung gerückt.

Dennoch ist es nach wie vor notwendig, jede einzelne gewünschte Funktionalität konkret zu implementieren. Dies widerspricht dem menschlichen Vorgehen, dem ein Streben nach der Automatisierung von (primitiven) Abläufen in der Geschichte viele Fortschritte brachte. So wäre es in der Kommunikation unter Mitmenschen undenkbar, dass Anweisungen bis ins kleinste Detail konkretisiert werden müssen. Keine Chefin würde zu ihrem Sekretär sagen: „Bitte notieren Sie: ‚Wo Verstand befiehlt, ist der Gehorsam

leicht.' – **Notieren Sie es im Notizbuch und nutzen Sie dazu einen Kugelschreiber!**"

Es wäre wünschenswert, auch eine Kommunikation mit dem Computer auf einer höheren Sprachebene zu vollziehen. Beispielsweise auf der Basis von Entitäten, die in der Lage sind, in einer Anweisungsliste notwendige Zwischenschritte selbständig zu interpolieren. Der Programmierer würde diesen möglicherweise kollaborativen, lernfähigen, proaktiven Entitäten seine Anweisungen übermittelt. Abgesehen von der Programmierung gibt es solche Entitäten bereits: Softwareagenten. Warum diese Technologie jedoch hier als wünschenswertes Hilfsmittel für das Programmieren proklamiert wird, das soll in einer Analogie abschließend dargestellt werden:

Wenn jemand ein Loch ausheben will – gräbt er mit den Fingern oder mit einem Stein? Auch wenn am Ende des Lochs ein Stein vergraben liegt, so ist es sinnvoll, zunächst nach einem Stein zum Graben zu suchen! Sind Softwareentwickler mit den herkömmlichen Programmierwerkzeugen den Anforderungen die die Entwicklung intelligenter Software stellt, gewachsen? Oder sollte man zunächst nach intelligenteren Werkzeugen streben um dieses Ziel zu erreichen? Die überschaubaren Fortschritte in der Entwicklung intelligenter Softwareagenten in der fast 30 jährigen Forschung zu diesem Thema weisen möglicherweise auf die Notwendigkeit solcher intelligenteren Werkzeuge hin.

Mit Agententechnologien können intelligentere Werkzeuge zur Erstellung von Software geschaffen werden. Die vorgestellten Konzepte der Softwareagenten kommen dabei zum Einsatz. Das agentenorientierte Programmieren, auch Agent Oriented Programming (AOP) genannt, ist ein viel versprechender Ansatz und könnte sich zu einem neuen Programmierparadigma entwickeln.

2.3. Begriff Personalisierung

Ähnlich dem Agentenbegriff wird auch der Begriff Personalisierung besonders in populärer Literatur oft ungenau eingesetzt. Da er jedoch wesentlich spezieller ist und damit besser fassbar, soll anders vorgegangen werden, als bei der Einführung des Begriffes Information.

In [Kluge, Menzel, 2002] wurde folgende Definition verwendet:

> *Personalisierung bedeutet die Anpassung von Inhalten für ein konkretes Subjekt, z.B. durch Hinzufügen von für dieses Subjekt interessanten und Weglassen von uninteressanten Informationen. Ein solches „konkretes Subjekt" könnte als eine reale Person oder auch als eine Gruppe von Personen verstanden werden.*

Zunächst stellt sich die Frage, warum Personalisierung notwendig ist. Dieser Frage soll an einem Beispiel nachgegangen werden:

Eine Mutter sagt zu ihrem Kind: „Würdest du bitte den großen roten Ordner mit den beiden Löchern aus dem Arbeitszimmer holen?". Das Kind hört sie nicht, worauf die Mutter ihren Mann bittet: „Würdest du mir bitte den Steuerordner bringen?"

In der Sprachwissenschaft nennt man dieses Verhalten Code-Switching – die Anpassung der Sprachform an den Zuhörer. Dies ist ein intuitiver Anwendungsfall der Personalisierung. Personalisierung kann also die Kommunikation unterstützen.

An dem Beispiel kann man außerdem feststellen, dass Personalisierung in der Kommunikation zwischen intelligenten Individuen auftreten kann, hier: der Familie. Personalisierung ist eine Anpassung von Kommunikationsinhalten an eine Person. „Person" ist hier jedoch abstrakt zu verstehen. Es kann sich dabei auch um einen Personenkreis oder gar um eine virtuelle Person handeln.

Wichtig ist eine gewisse Persönlichkeit, d.h. individuelle Merkmale, nach denen die Anpassung vorgenommen werden kann.

Im WWW wird die Personalisierung meist in zwei Formen sichtbar:

- als Anpassung des Interfaces von Webseiten: z.B. durch das Anzeigen von Shortcuts oder durch das Weglassen von Links, die vom Nutzer nicht benutzt werden
- als Filterung von Informationen: z.B. durch Weglassen von Nachrichten auf der Startseite eines Nachrichtenportals, von denen angenommen werden kann, dass der Nutzer daran nicht interessiert ist

Während die Gestaltung des Interfaces von Webseiten und die Auswahl der Informationen auf Webseiten in herkömmlichen Web-Anwendungen an den Bedürfnissen einer gesamten Zielgruppe ausgerichtet werden, so ermöglicht die Personalisierung eine Rücksichtnahme auf die Bedürfnisse einzelner Nutzer.

2.3.1. Sicherheitsaspekte und Datenschutz

Unter vielen Anwendern ist eine gewisse Skepsis gegenüber der Personalisierung festzustellen. Dies hängt zum einen oft mit Unwissenheit zusammen und daraus resultierenden Vorurteilen bezüglich der Sicherheit und Diskretion. Zum anderen fühlen sich besonders fortgeschrittene Anwender gelegentlich von zum Teil unausgereiften Anwendungen der Personalisierung belästigt, wie den bekannten Microsoft Assistenten. Erfolgreiche Personalisierungen bleiben für Laien jedoch oft unbemerkt, z.B. auf der Website des Online-Händlers Amazon.

Besorgnis um den Anonymitätsverlust oder das ungewollte Preisgeben persönlicher Eigenschaften sind natürlich ernst zu nehmen und begründet. Zu einer pauschalen Verurteilung der Personalisierung sollten sie jedoch nicht

führen. Die Gewährleistung der Sicherheit persönlicher Daten ist eine Frage der konkreten Implementation. Wie auch bei anderen sensiblen Vorgängen, etwa bei Zahlungsvorgängen, ist eine Einschätzung der Sicherheit ohne eine Untersuchung der zugrunde liegenden Technik nur beschränkt möglich. Es bleibt für den Endkunden also eine Frage des Vertrauens in den Anbieter einer Web-Dienstleistung. Dieser kann sich einer freiwilligen Sicherheitskontrolle unterziehen, z.B. dem TÜV für IT-Security, um Vertrauen aufzubauen.

Es kann jedoch generell gesagt werden, dass vielen Methoden der Personalisierung ein sehr abstrakter Datenbestand zugrunde liegt. Diese Daten lassen ohne Kenntnisse der konkreten Methoden meist keine Rückschlüsse auf natürliche Eigenschaften einer Person zu.

In [Kluge, Menzel, 2002] wurde ein Personalisierungskonzept vorgestellt, welches eine Unabhängigkeit zwischen dem Nutzer innerhalb der Anwendung und dem Nutzer innerhalb des Personalisierungssystems ermöglicht. Konkrete Daten zum Nutzer, die der Anwendung vorliegen, z.B. Name und Anschrift, sind nicht mit den Daten in Verbindung zu bringen, die dem Personalisierungssystem vorliegen. Dies kann ein Sicherheitsgewinn sein und ermöglicht außerdem die Personalisierung von Systemen, die keine eigene Nutzerverwaltung haben.

Datenschutz

Die Datenschutzthematik spielt dann eine Rolle, wenn personenbezogene Daten verarbeitet werden. *"Personenbezogene Daten sind Einzelangaben über persönliche oder sachliche Verhältnisse einer bestimmten oder bestimmbaren natürlichen Person (Betroffener)."*[3] Wie im vorigen Absatz beschrieben, ist das bei Personalisierungen nicht zwangsläufig der Fall. Gibt es keine Verbindung zwischen einem abstrakten Nutzer, dessen Navigationsverhalten beispielsweise

[3] § 3 Bundesdatenschutzgesetz

für die Personalisierung analysiert wird, und einem realen Nutzer, dessen Name und Anschrift bekannt ist, dann handelt es sich bei den Navigationsdaten nicht um personenbezogene Daten.

Rechtliche Grundlagen

Das Bundesdatenschutzgesetz (BDSG)[4] regelt die rechtlichen Grundlagen des Datenschutz in Deutschland. Teledienstegesetz (TDG)[5], Teledienstedatenschutzgesetz (TDSSG)[6] und der Mediendienstestaatsvertrag (MDStV)[7] sind in rechtlichen Aspekten von Internetdienstleistungen außerdem relevant. Eine Grundlage der datenschutzrechtlichen Gesetzgebung ist die Informationelle Selbstbestimmung (ISR). Dieser Begriff wurde im Rahmen eines Urteils des Bundesverfassungsgerichtes 1983 erstmals geprägt. Ein Bürger kann in seiner Freiheit wesentlich gehemmt sein, wenn für ihn nicht überschaubar ist, welche Informationen seiner Umwelt über ihn bekannt sind, wurde in der Gesetzgebung argumentiert.

Die wesentlichen gesetzlichen Vorgaben, die für ein personalisierendes System relevant sind, lauten:

- Datensparsamkeit: Es dürfen nur Daten verarbeitet werden, die für den konkreten Fall erforderlich sind.
- Zweckbindung: Die Datensammlung hat an einen Zweck gebunden zu sein, pauschale Datensammlung ist nicht zulässig.
- Löschung: Betroffene Personen können die Berichtigung, Löschung oder Sperrung von Daten fordern.

[4] http://www.datenschutz-berlin.de/recht/de/bdsg/bdsg1.htm (abgerufen am 1. Juli 2003)
[5] http://www.netlaw.de/gesetze/tdg.htm (abgerufen am 1. Juli 2003)
[6] http://www.netlaw.de/gesetze/tddsg.htm (abgerufen am 1. Juli 2003)
[7] http://www.datenschutz-berlin.de/recht/de/stv/mdstv.htm (abgerufen am 1. Juli 2003)

- Auskunft: Ein Betreiber hat auf Anfrage Auskunft zu geben, welche personenbezogenen Daten verarbeitet werden und zu welchem Zweck dies geschieht.

- Transparenz: Die Datenverarbeitung muss nachvollziehbar organisiert sein; dazu gehört auch eine Benachrichtigung der Personen vor der Erhebung der Daten.

Es gibt weiterhin persönliche Angaben, die schärferen Bestimmungen unterliegen. Dazu Zählen Angaben zur rassischen und ethnischen Herkunft, politischen Meinung, religiösen oder philosophischen Überzeugung, Gewerkschaftszugehörigkeit, Gesundheit oder Sexualleben.

Im § 46 des BDSG werden Anforderungen an die Organisation der Datenverarbeitung genannt:

> *"1. Unbefugten den Zutritt zu Datenverarbeitungsanlagen, mit denen personenbezogene Daten verarbeitet oder genutzt werden, zu verwehren (Zutrittskontrolle),*
>
> *2. zu verhindern, dass Datenverarbeitungssysteme von Unbefugten genutzt werden können (Zugangskontrolle),*
>
> *3. zu gewährleisten, dass die zur Benutzung eines Datenverarbeitungssystems Berechtigten ausschließlich auf die ihrer Zugriffsberechtigung unterliegenden Daten zugreifen können, und dass personenbezogene Daten bei der Verarbeitung, Nutzung und nach der Speicherung nicht unbefugt gelesen, kopiert, verändert oder entfernt werden können (Zugriffskontrolle),*
>
> *4. zu gewährleisten, dass personenbezogene Daten bei der elektronischen Übertragung oder während ihres Transports oder ihrer Speicherung auf*

Datenträger nicht unbefugt gelesen, kopiert, verändert oder entfernt werden können, und dass überprüft und festgestellt werden kann, an welche Stellen eine Übermittlung personenbezogener Daten durch Einrichtungen zur Datenübertragung vorgesehen ist (Weitergabekontrolle),

5. zu gewährleisten, dass nachträglich überprüft und festgestellt werden kann, ob und von wem personenbezogene Daten in Datenverarbeitungssysteme eingegeben, verändert oder entfernt worden sind (Eingabekontrolle),

6. zu gewährleisten, dass personenbezogene Daten, die im Auftrag verarbeitet werden, nur entsprechend den Weisungen des Auftraggebers verarbeitet werden können (Auftragskontrolle),

7. zu gewährleisten, dass personenbezogene Daten gegen zufällige Zerstörung oder Verlust geschützt sind (Verfügbarkeitskontrolle),

8. zu gewährleisten, dass zu unterschiedlichen Zwecken erhobene Daten getrennt verarbeitet werden können."

2.3.2. Überblick über Personalisierungstechniken

Im folgenden Abschnitt sollen Personalisierungstechniken vorgestellt werden, die in kommerziellen Anwendungen bereits erfolgreich eingesetzt werden.

Regelbasierte Verfahren

In regelbasierten Verfahren werden von Experten (Experten bezüglich der zu personalisierenden Anwendung) Regeln formuliert, die den Zusammenhang

zwischen den über den Nutzer gesammelten Informationen und der Anwendung beschreiben. Eine solche Regel kann wie folgt aussehen:

```
Hat der Nutzer Interesse an Inhalten zum Thema Sport, dann
zeige auch Inhalte zum Thema Aktivurlaub.
```

Beispiel 3: Regel in einer regelbasierten Personalisierung (natürlichsprachlich)

Solche Regeln werden oft auf Basis von prädikatenlogischen Formulierungen ausgedrückt, in einem wissensverarbeitenden System verarbeitet und mittels Inferenzmaschine (enthält den Schlussfolgerungsmechanismus) und Wissensbasis (enthält neben den Regeln auch die Fakten über die Nutzer) ausgewertet.

Ein Nachteil dieser Personalisierungstechnik ist der hohe Initial- und Wartungsaufwand im Vergleich zu statistischen Verfahren der Personalisierung. Jede Regel muss genau bestimmt werden und selbst geringfügig von den Regeln abweichende Situationen werden durch die Regeln nicht mehr abgedeckt. Ändert sich die Umgebung, innerhalb derer die Regeln definiert sind, so ist eventuell eine Anpassung der betroffenen Regeln notwendig. Im Beispiel 3 wäre dies der Fall, wenn die Kategorie *Sport* durch zwei Kategorien: *Sport*, *Freizeit* ersetzt würde. In diesem Fall sollte die Regel auch auf die neue Kategorie *Freizeit* ausgedehnt werden.

Ein Vorteil dieser Personalisierungstechnik ist die Genauigkeit und Steuerbarkeit durch den Betreiber. Durch entsprechendes Formulieren der Regeln kann der Betreiber die Personalisierungsziele unmissverständlich bestimmen und somit exakt kalkulieren. Dass dies nicht selbstverständlich ist, kann man bei der Betrachtung weiterer Personalisierungstechniken feststellen.

Nutzergestützte Verfahren

In nutzergestützten Personalisierungstechniken werden die über die Nutzer gesammelten Daten gewissen Filtertechniken unterzogen. Diese Filter können sehr einfach sein (z.B. Premium-Nutzer dürfen Premium-Inhalte betrachten) aber auch auf komplexen statistischen Berechnungen basieren (z.B. Korrelationskoeffizient nach Pearson). Zu den fortgeschritteneren Filtertechniken in der nutzergestützten Personalisierung gehören:

- **Inhaltsbasiertes Filtern** (auch Cognitive/Content Based Filtering genannt): Diese Filtertechnik basiert auf der Zuordnung von Attributen zu den Inhalten einer Anwendung. Greift ein Nutzer auf diese Inhalte zu, so kann anhand der Attribute sein persönliches Profil beeinflusst werden. Soll auf personalisierte Inhalte zurückgegriffen werden, so kann umgekehrt auch das Profil des Nutzers mit den Attributen der Inhalte verglichen werden, um nur passende Inhalte zu wählen. Voraussetzung für diese Form des Filterns ist allerdings eine ausreichend distinkte Klassifikation der Inhalte. Sind die Inhalte nicht klar trennbar, so ist die Zuordnung aussagekräftiger Attribute schwieriger, die Qualität der Personalisierung damit geringer.

- **Kollaboratives Filtern**: Viele Web-Anwendungen erlauben Rückschlüsse auf das Interesse eines Nutzers an Inhalten der Anwendung. Dieses Interesse kann ein Nutzer durch den Kauf der Inhalte kundtun, durch den Aufenthalt auf Inhaltsseiten, durch das Klicken auf Links oder ähnliches. Anhand des Vergleiches der Bewertungsmuster verschiedener Nutzer ist es möglich, Nutzergruppen zu bilden. Nutzer, die einer Gruppe angehören, haben ein ähnliches Bewertungsmuster und somit ähnliche Interessen bezüglich der Inhalte der Anwendung. Es existiert eine Schnittmenge der Interessen dieser Nutzer (Abbildung 7). In den meisten Fällen existieren jedoch auch Differenzmengen, da die Interessen

der Nutzer nicht identisch sind. In der Praxis hat sich gezeigt, dass unter bestimmten Kriterien (bei ausreichend großer Schnittmenge, bei geeigneten Inhalten u. a.) angenommen werden kann, dass die Vereinigungsmenge die Interessen der einzelnen Nutzer qualitativ darstellt. Nach Abbildung 7 ermöglicht dies, für Nutzer a auch Inhalte personalisiert vorzuschlagen, für die er keine Bewertung abgegeben hat, nämlich die Menge „vermutetes Interesse von Nutzer a“.

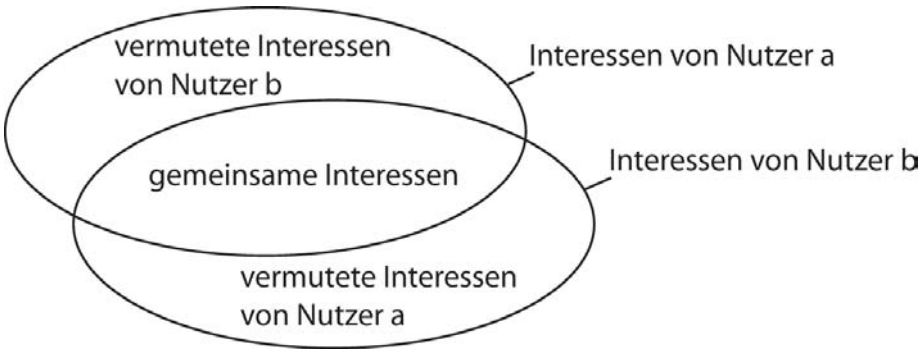

Abbildung 7: gemeinsame und unterschiedliche Interessen zweier Nutzer für die Realisierung eines kollaborativen Filterns

3. Konzeption

Nachdem die Grundbegriffe und die Grundlagen erläutert wurden, soll nun auf das konkrete Konzept zum Entwurf eines personalisierenden Suchagenten eingegangen werden.

Eine Implementation wird in dieser Arbeit nicht verfolgt. Nach dem klassischen Vorgehensmodell der Softwareentwicklung handelt es sich um Machbarkeitsstudie, Anforderungsanalyse und einen Systementwurf. Da jedoch keine Software entwickelt werden soll, geschieht dies jeweils auf abstraktem Niveau.

3.1. Machbarkeit

Soll ein Suchwerkzeug für das WWW konzipiert werden, so kann dabei bis auf ein sehr abstraktes Level zurückgegangen werden. Bei der Betrachtung des Grundbegriffes *Information* in Kapitel 2.1 wurde dies deutlich. Da im Rahmen dieser Arbeit nur ein begrenzter Aufwand betrieben werden kann, sollen einige Einschränkungen festgelegt werden:

Datenquelle

Als Informationsquelle soll auf die Datenbanken herkömmlicher Suchmaschinen zurückgegriffen werden[8].

Dies bringt einen entscheidenden Vorteil: das Konzept kann bei dem Suchwerkzeug an sich beginnen, die Datenquellen werden als gegeben betrachtet. Initialisierung und Aktualisierung der Datenquellen sind somit nicht Bestandteil dieses Konzeptes.

[8] als herkömmliche Suchmaschinen seien hier Google, MSN, Hotbot oder ähnliche Konzepte bezeichnet

Ein Nachteil ist zunächst, dass das Suchwerkzeug nur so mächtig sein kann, wie die Gegebenheiten der Datenquellen dies zulassen. Während bei der Erstellung einer eigenen Datenbasis mächtigere Konzepte verwendet werden könnten als die der Volltext-Indizierung herkömmlicher Suchmaschinen, so ist das Suchwerkzeug in diesem Fall auf die Gegebenheiten der Datenquellen und Schnittstellen beschränkt. Es wird dennoch deutlich werden, dass trotz der beschränkten Möglichkeiten, die die Datenquellen herkömmlicher Suchmaschinen bieten, fortgeschrittenere Eigenschaften realisierbar sind.

Interface

Es soll an dieser Stelle darauf hingewiesen werden, dass eine optimale Schnittstelle zur Kommunikation zwischen Suchwerkzeug und Benutzer durchaus nicht dem bekannten Schema: „Suchanfrage in Form von Stichworten – Suchausgabe in Form von Links und kurzen Erläuterungen“ entsprechen muss. Andere Konzepte sollen jedoch nicht untersucht werden.

Bei der Betrachtung der Anforderungen und des Systementwurfs in den nächsten Kapiteln wird deutlich werden, dass das Suchwerkzeug clientseitig ausgeführt werden muss. Um das Verständnis der folgenden Ausführungen zu fördern, sei ein prototypisches Interface vorgestellt:

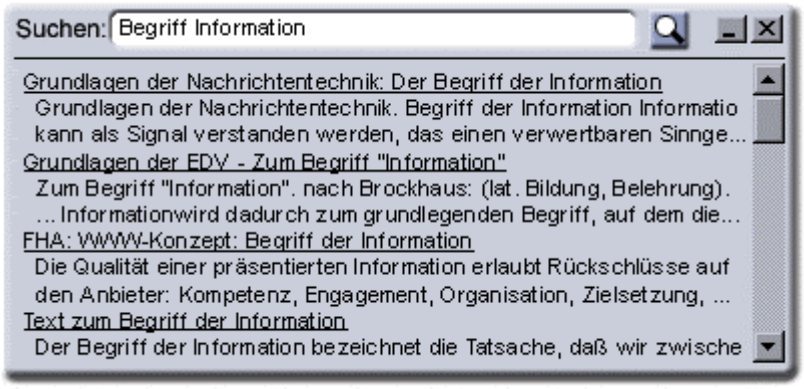

Abbildung 8: prototypisches Interface des konzipierten Suchwerkzeuges

Agententechnologien und Personalisierung

Der Fokus des Konzeptes liegt auf der Auswahl und dem Zusammenspiel von Agententechnologien und Methoden der Personalisierung. Es werden außerdem Verfahren der Computerlinguistik zum Einsatz kommen.

Softwareagenten und Personalisierungstechniken wurden bereits in Kapitel 2 besprochen und in zahlreichen Studien und Anwendungen eingesetzt. Auch die verwendeten Verfahren der Computerlinguistik konnten in diversen Anwendungen erfolgreich genutzt werden. Eine prinzipielle Machbarkeit der Bestandteil, auf denen das Konzept aufbaut, muss also nicht weiter untersucht werden.

Hypothese

Die Hypothese, auf der das Konzept basiert, ist folgende:

Auch unter Verwendung der Datenquellen herkömmlicher Suchmaschinen (z.B. Google) und unter Beschränkung auf herkömmliche Suchmaschinen-Interfaces (Eingabe eines Suchtextes; Ausgabe einer Linkliste) ist die Konzeption eines Suchwerkzeuges möglich, welches *subjektiv bessere* Suchergebnisse liefert.

Im Rahmen dieser Arbeit wird diese Hypothese nicht bewiesen. Ob Suchergebnisse subjektiv besser sind, kann nur empirisch nachgewiesen werden. Eine Implementation, und damit die Voraussetzung für eine empirische Studie, wird in dieser Arbeit nicht beschrieben. Es soll jedoch deutlich werden, dass das Konzept das Potential dazu bietet.

3.2. Anforderungen

Welche Wünsche kann ein Nutzer an ein Werkzeug zum Suchen von Informationen im Internet haben? Diese Fragestellung ist natürlich sehr allgemein. Unter Berücksichtigung der in Kapitel 3.1 erklärten Beschränkungen können folgende Anforderungen formuliert werden:

- einfache, flexible und mächtige Sprache zur Formulierung von Suchanfragen
- Schnelligkeit
- Aktualität
- geringer Bedienungsaufwand
- Anpassungsfähigkeit (an die Interessen des Nutzers)
- Zufrieden stellende Suchergebnisse

Suchanfragen

In den meistgenutzten Suchmaschinen[9] werden Suchanfragen in Form von Stichwörtern formuliert:

```
"Häuser Leipzig Gründerzeit"
```

Beispiel 4: Suchanfrage in Stichwortform

Mittels einer Suchanfrage soll ein Interessengebiet spezifiziert werden. Die **Aufzählung** von Stichwörtern (Beispiel 4) ermöglicht lediglich die Konkretisierung eines Interessengebietes: im Beispiel soll nicht nur nach Informationen zu *Häusern*, zu *Häusern und Leipzig*, sondern zu *Häusern, Leipzig*

[9] 1. Google; 2. Yahoo; 3. MSN; 4. AOL (US-Markt, Januar 2003; nach http://searchenginewatch.com/reports/article.php/2156451, abgerufen am 2.7.2003)

und Gründerzeit gesucht werden. Es wird also nach Websiten gesucht, von denen diese drei Themen behandelt werden (bzw. in denen diese drei Wörter auftauchen). Tatsächlich durchsuchen die meistgenutzten Suchmaschinen Dokumente nicht nach Themen oder Inhalten, sondern nach Wörtern auf syntaktischer Ebene. Zwar kann auf dieser Ebene auf einfache Tippfehler hingewiesen werden, wie dies bei Google der Fall ist, aber es können keinerlei semantische Konzepte zum Einsatz kommen. So ist es nicht möglich, eine Verbindung zwischen dem Eigennamen „Wolfgang Tiefensee“ und dem Text „Leipziger Bürgermeister“ herzustellen, da die Systeme nicht auf semantischer Ebene agieren.

Dennoch erlaubt die Stichwortsuche mehr als eine Aufzählung von Stichwörtern. In Tabelle 1 werden Operatoren der drei meistgenutzten Suchmaschinen aufgelistet (AOL nutzt Google und ist deshalb nicht in der Liste enthalten). Mit Hilfe dieser Operatoren können auch **Phrasen** und **Boolsche Ausdrücke** als Suchanfragen formuliert werden. MSN unterstützt weiterhin den Einsatz von **Platzhaltern** (Wildcards).

	Beispiel	Google	Yahoo	MSN
UND-Verknüpfung	Leipzig UND Bürgermeister	ja	ja	ja
ODER-Verknüpfung	Leipzig ODER Bürgermeister	ja	ja	ja
NICHT-Verknüpfung	Leipzig NICHT(Bürgermeister)	ja	ja	ja
Phrasen	„Leipzig Bürgermeister“	ja	ja	ja
Platzhalter	Leipzig B*rgermeister	nein	nein	ja

Tabelle 1: Suchfunktionen auf syntaktischer Ebene der 3 meistgenutzten Suchmaschinen

Diese Operatoren ermöglichen lediglich die Selektion von Daten. Es handelt sich demnach nicht um Abfragesprachen, denn die Selektion ist nur ein

Bestandteil einer Abfragesprache. Es besteht die Hoffnung, dass dies den technischen Umgang (z.B. Analyse oder Manipulation) mit der Suchanfrage vereinfacht.

Es stellt sich die Frage, ob diese Form der Suche für den Nutzer ausreichend einfach, flexibel und mächtig ist. Eine Studie der Univertität Waikato [Jones et al., 1998] untersuchte Suchanfragen der Suchmaschine des Computer Science Technical Report (CSTR) über einen Zeitraum von 61 Wochen. Es wurden 30.000 Suchanfragen aufgezeichnet und ausgewertet, wobei die folgenden Schlussfolgerungen gezogen wurden konnten:

- **Standardeinstellungen**: Nutzer tendieren dazu, Standardeinstellungen zu akzeptieren, unabhängig davon, ob diese ihren Wunscheinstellungen entsprechen. Zu den Einstellungsmöglichkeiten gehörte der Anfragetyp (ranked – Suchergebnisse werden nach Relevanz sortiert / boolsch – keine Sortierung nach Relevanz), die Standardverknüpfung (UND / ODER), die Groß-/Kleinschreibung und andere.
- **Komplexität der Suchanfragen**: Suchanfragen sind meist einfach und kurz. Durchschnittlich wurden 2,5 Terme verwendet; über 80% der Suchanfragen enthielten ein, zwei oder drei Terme. Ein Viertel der Anfragen enthielt mindestens eine UND-Verknüpfung, ODER-Verknüpfungen, NICHT-Verknüpfungen und Phrasen wurden zu jeweils unter 5% verwendet. Dabei ist zu beachten, dass die Zielgruppe des CSTR mit diesen Operatoren vertrauter sein dürfte als die Anwender populärer Suchmaschinen dies sind. Die Mächtigkeit boolscher Operatoren für die Formulierung einer Suchanfrage ist groß, dennoch führen einige Faktoren dazu, dass diese nur wenig genutzt werden. Zu diesen Faktoren gehören der unterschiedliche Syntax in

vielen Suchmaschinen und die widersprüchliche Verwendung der Ausdrücke UND sowie ODER in der Umgangssprache im Gegensatz zur Verwendung in der boolschen Logik (ODER meint in der Umgangssprache oft ein exklusives ODER, in der boolschen Logik ein nicht-exklusives ODER).

- **Verfeinerung der Suchanfragen**: In 66% der fortlaufenden Suchanfragen (erneute Anfrage von Nutzern, die in dieser Session bereits eine Anfrage gestellt haben) ist mindestens einen Term der Suchanfrage identisch mit der vorigen Anfrage. In jeweils 23% der Fälle sind ein oder zwei Terme identisch. Gemeinsam mit weiteren beobachteten Gegebenheiten lässt sich daraus vermuten, dass Nutzer ihre initialen Suchanfragen verfeinern. Dies kann darauf hinweisen, dass die Nutzer anhand des Suchergebnisses realisierten, dass die Suchanfrage ihren eigentlichen Suchwunsch nicht erwartungsgemäß repräsentiert.

Zusammenfassend kann folgendes festgestellt werden:

Es ist davon auszugehen, dass Standardeinstellungen bezüglich der Suchanfragen nur von wenigen Nutzern geändert werden. Demnach sollten diese den Erwartungen der Nutzer entsprechen. Bedienelemente für diese Einstellungen müssen keine hervorragende Position im Interface einnehmen.

Komplexe, proprietäre Sprachen zur Formulierung von Suchanfragen sind für ein Suchwerkzeug mit nicht spezieller Zielgruppe nicht sinnvoll.

Es kann eine Diskrepanz zwischen der natürlichen Sprache und der Sprache, in der die Suchanfragen formuliert werden, geben. Dies äußert sich darin, dass ein Nutzer eine Suchanfrage nachträglich verfeinern muss, weil er anhand des Suchergebnisses feststellt, dass die Suchanfrage nicht seinem eigentlichen Suchwunsch ausdrückt.

Aufgrund der in Kapitel 3.1 aufgeführten Einschränkungen wird das Suchwerkzeug auf eine Sprache zur Formulierung von Suchanfragen beschränkt sein, die sich auf die Selektion von Daten beschränkt. Wie es von den meistgenutzten Suchmaschinen bekannt ist, besteht eine solche Suchanfrage aus Termen (Suchbegriffen) und Operatoren. Soll das Suchwerkzeug auf mehrere Suchmaschinen zugreifen können (wie eine Metasuchmaschine), so ist bei der Spezifikation der Sprache darauf zu achten, dass die verwendeten Operatoren von den Suchmaschinen interpretiert werden können, oder dass die Suchanfrage so umgeformt werden kann, dass sie nur gültige Operatoren der Suchmaschinen enthält. Dies ist mit dem Platzhalter-Operator der Suchmaschine MSN (siehe Tabelle 1) nicht möglich.

Da komplexe Operatoren nicht bereitgestellt werden müssen (siehe Punkt 2 des letzten Abschnitts), soll es genügen, die boolschen Verknüpfungsmöglichkeiten in der Sprache zu implementieren. Diese werden von allen meistgenutzten Suchmaschinen unterstützt (siehe Tabelle 1).

Schnelligkeit

Die Schnelligkeit eines Suchwerkzeuges wird sich an der Schnelligkeit vergleichbarer Werkzeuge messen lassen müssen. Vergleichbar sind in diesem Fall herkömmliche Suchmaschinen, z.B. Google. Dennoch hängt die Geduld des Anwenders von einem konkreten Anwendungsfall ab. Dabei spielt z.B. auch die erwartete Qualität des Suchergebnisses eine Rolle.

Aktualität

Herkömmliche Suchmaschinen bilden Teile des Internets in einer internen Datenbank ab. Dies ermöglicht das Suchen von Informationen in der Geschwindigkeit, wie man es von diesen Suchmaschinen gewöhnt ist. Da es

sich in der lokalen Datenbank jedoch nur um eine Abbildung der tatsächlichen Informationsquellen handelt (z.B. Webseiten), kann es vorkommen, dass diese Abbildung nicht mehr up-to-date ist. Diese Aktualität ist keine Frage dieses Konzeptes, da die Datenbasen als gegeben betrachtet werden.

Bedienungsaufwand

Die Bereitschaft des Anwenders, Änderungen an den Einstellungen des Suchwerkzeuges vorzunehmen, ist als gering einzuschätzen (siehe Seite 3-62). Die Bedienungsoberfläche sollte dies widerspiegeln, indem die Bedienelemente auf das Nötigste begrenzt werden. Dies fördert die Übersichtlichkeit und damit die Usability. Im Zentrum der Bedienung steht die Übermittlung eines Suchwunsches durch den Anwender. Es ist an dieser Stelle bewusst von *Übermittlung* die Rede, da dies nicht zwangsläufig *Eingabe* bedeutet. Ein Suchwunsch kann auch aus Aktionen eines Anwenders abgeleitet werden, wie beispielsweise dem Schreiben einer wissenschaftlichen Arbeit. Eine automatische Analyse dieses Textes, während er vom Anwender verfasst wird, ist durchaus denkbar. Ergebnis dieser Analyse kann eine Suchanfrage sein, die den aktuellen Informationskontext des Anwenders widerspiegelt. Bei der Konzeption sollen die Möglichkeiten automatischer Suchanfragen Beachtung finden.

Anpassungsfähigkeit

Wie bereits bei der Betrachtung von Informations-/Internetagenten in Kapitel 2.2.3 erwähnt, machen Art und Häufigkeit der Nutzung von Suchwerkzeugen eine Anpassungsfähigkeit bzw. Lernfähigkeit des Suchwerkzeuges gegenüber dem Nutzer gut möglich. Diverse Verfahren können verwendet werden, um eine Personalisierung des Suchwerkzeuges zu ermöglichen. So kann

beispielsweise die Beurteilung der Relevanz einer Seite nach dem Interessenprofil des Anwenders vorgenommen werden.

Zufriedenstellende Suchergebnisse

Kann ein Suchwerkzeug alle aufgeführten Anforderungen erfüllen, so sind zunächst nur die Rahmenbedingungen geschaffen. Schließlich muss die Qualität der Suchergebnisse den Erwartungen des Anwenders entsprechen. Die Erwartung ist eine subjektive Einschätzung und orientiert sich sowohl an den Erfahrungen mit anderen Suchwerkzeugen als auch an dem Aufwand, der mit der Initialisierung eines Suchvorgangs verbunden ist. Muss das Suchwerkzeug lokal installiert werden, so ist davon auszugehen, dass die Erwartungen des Anwenders über denen liegen, die er einer herkömmlichen Suchmaschine entgegenbringt. Diesem Anspruch kann eventuell auch durch Vorzüge in den sonstigen aufgeführten Anforderungen Genüge getan werden (z.B. bessere Bedienbarkeit, Anpassungsfähigkeit). Es besteht jedoch die Hoffnung, dass aufgrund der erweiterten Möglichkeiten, die sich diesem Konzept gegenüber einer herkömmlichen Suchmaschine bieten, auch die Qualität der Suchergebnisse als besser empfunden werden kann.

3.3. Systementwurf

Nachdem festgestellt wurde, was ein Suchwerkzeug leisten muss, kann man sich Gedanken darüber machen, wie es das leisten soll.

Dazu soll das Problem zunächst in Teilprobleme untergliedert werden. Wie oft in der Datenverarbeitung bietet es sich an, zwischen Eingabe, Verarbeitung und Ausgabe von Daten zu unterscheiden.

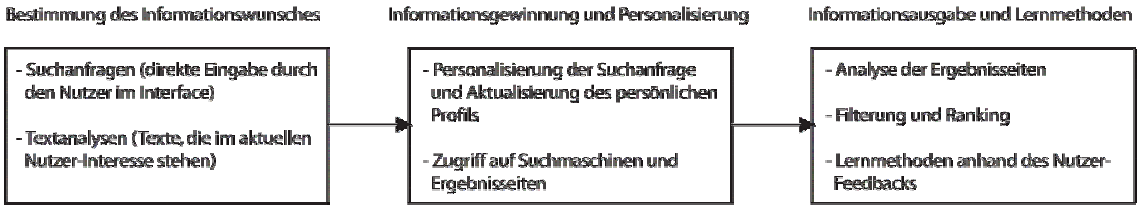

Abbildung 9: Schematischer Programmablauf in drei Phasen

Der Programmablauf wird in drei Phasen untergliedert (Abbildung 9), die in den folgenden drei Kapiteln erläutert werden. Um einen Überblick zu ermöglichen, sollen die Aufgaben der einzelnen Phasen hier bereits kurz angeführt werden:

- **Phase 1 – Bestimmung des Informationsbedürfnisses:** Ein Informationsbedürfnis besteht nicht nur dann, wenn ein Nutzer diesen direkt in das Interface des Suchwerkzeuges in Form einer Suchanfrage eingibt. Im Abschnitt Bedienungsaufwand auf Seite 3-62 wurde bereits darauf aufmerksam gemacht, dass es wünschenswert wäre, wenn ein Suchwerkzeug das Informationsbedürfnis des Nutzers selbständig erkennt. In Kapitel 3.3.1 werden Methoden beschrieben, die die Kommunikation zwischen Anwender und Suchwerkzeug auf einer höheren Ebene vollziehen als auf der einer bloßen Stichworteingabe in herkömmlichen Suchmaschinen.

- **Phase 2 – Informationsgewinnung und Personalisierung:** Nachdem das Informationsbedürfnis in eine Suchanfrage überführt wurde, kann diese Suchanfrage an Suchmaschinen gestellt werden. Des weiteren kann anhand der Suchanfrage das persönliche Profil des Nutzers beeinflusst werden, d.h. die Personalisierungsdaten können aktualisiert werden. Umgekehrt kann auch die Suchanfrage aufgrund des Profils des Nutzers erweitert oder manipuliert werden. Die von den Suchmaschinen vorgeschlagenen Dokumente werden geladen. In Kapitel 3.3.2 wird Phase 2 erläutert.

- **Phase 3 – Informationsausgabe und Lernmethoden:** Die Ergebnisseiten werden analysiert und deren Relevanz für den Nutzer anhand der Suchanfrage sowie des Profils des Nutzers beurteilt. Aus dem Feedback des Nutzers, z.B. aus der Wahl des vom Suchwerkzeug vorgeschlagenen Dokumentes, können Schlussfolgerungen gezogen werden, die die zukünftige Qualität des Suchwerkzeuges verbessern sollen. Auf Phase 3 wird in Kapitel 3.3.3 eingegangen.

3.3.1. Phase 1 – Bestimmung des Informationsbedürfnisses

Es soll hier zwischen zwei Verfahren unterschieden werden, das Informationsbedürfnis des Anwenders zu bestimmen:

- direktes Informationsbedürfnis – der direkten Formulierung einer Suchanfrage durch den Anwender
- indirektes Informationsbedürfnis – der automatischen Ermittlung des Informationsbedürfnisses

3.3.1.1. Direktes Informationsbedürfnis

Der einfachste Fall für ein Suchwerkzeug, das Informationsbedürfnis des Nutzers zu ermitteln, ist der, dass der Nutzer dieses direkt dem Suchwerkzeug

mitteilt. Direkt bedeutet im Fall herkömmlicher Suchwerkzeuge die textuelle Eingabe. Da diese von dem Nutzer verlangen, den Suchwunsch in der jeweiligen Sprache des Suchwerkzeuges zu formulieren, kann diese Formulierung gemäß Abbildung 9 unverändert an das Modul *Informationsgewinnung und Personalisierung* weitergegeben werden. Die Bestimmung des Informationswunsches ist für den Fall der direkten Eingabe damit abgeschlossen.

3.3.1.2. Indirektes Informationsbedürfnis

Ein Vorteil eines Client-seitigen Suchwerkzeuges ist der, dass die Schnittstelle zum Anwender nicht so beschränkt werden muss, wie bei einer Web-Anwendung[10]. So kann ein Client-seitiges Programm dergestalt konzipiert sein, dass es ähnlich einem Interfaceagenten die Arbeitsweise eines Nutzers beobachtet, um dem Nutzer bei der weiteren Arbeit unterstützend zur Seite zu stehen. Um die Idee der automatischen Ermittlung eines Informationsbedürfnisses des Nutzers verständlich zu machen, soll hier ein Beispiel betrachtet werden.

Beispiel: Ermittlung eines indirekten Informationsbedürfnisses

Angenommen ein Anwender schreibt eine wissenschaftliche Arbeit mit einem Textverarbeitungsprogramm. In vielen Fällen, so ist anzunehmen, besteht/entsteht während des Schreibens ein Informationsbedürfnis. Dieses Informationsbedürfnis ist kontextbezogen, es hängt vom aktuellen Kontext des Anwenders ab. In diesem konkreten Fall wird der Kontext durch den vom Anwender bisher formulierten Text repräsentiert. Auch andere Aktivitäten wie das Formulieren von Suchanfragen oder das Lesen von Dokumenten können mit der Aktion und somit mit dem Informationsbedürfnis in Verbindung

[10] unter Web-Anwendung sie hier eine Client-Server Internetanwendung auf HTML-Basis (Client-seitig) und Webserver-/CGI-Programm-Basis (Server-seitig) verstanden

gebracht werden. Angenommen ein Suchwerkzeug ist in der Lage, diesen Kontext auf eine Suchanfrage automatisch abzubilden. Dazu kann der Kontext zunächst auf textuelle Informationen beschränkt werden. Mit geeigneten Filtermethoden werden sowohl aktive Eingaben (in diesem Beispiel: die wissenschaftliche Arbeit des Anwenders) als auch vom Anwender gelesene Texte auf eine Suchanfrage abgebildet. *Abbilden* heißt in diesem Fall, eine Suchanfrage zu formulieren, die die Texte informell möglichst genau repräsentiert.

Diese Suchanfrage kann vom Suchwerkzeug wie ein direkt vom Anwender formuliertes Informationsbedürfnis behandelt werden und zur Verarbeitung an das entsprechende Modul (Abbildung 9) weitergeleitet werden. Schließlich werden dem Anwender noch während des Schreibens seiner wissenschaftlichen Arbeit im Fenster des Suchwerkzeuges eine Liste von Dokumentenlinks präsentiert, die seinem aktuellen Informationsbedürfnis entsprechen.

Zum einen wird dem Anwender damit die direkte Eingabe von Suchanfragen erspart. Er kann den Fokus auf seiner eigentlichen Aufgabe belassen, nämlich dem Thema, mit dem er sich auseinandersetzt, und nicht der Bedienung eines Suchwerkzeuges. Zum anderen kann von Anwendern nicht erwartet werden, dass diese Experten auf dem Gebiet der Informationsbeschaffung sind; auch wenn die Bedienung einer herkömmlichen Suchmaschine trivial ist, so ist das Formulieren einer Suchanfrage, die dem persönlichen Informationsbedürfnis möglichst nahe kommt, durchaus nicht trivial. In [Jones et al., 1998] wurde festgestellt, dass nur 64% der Nutzer nach einer Suchanfrage tatsächlich auf eines der vorgeschlagenen Dokumente klickt. Die automatische Generierung von Suchanfragen wäre eine Möglichkeit, sowohl Bedienung als auch Qualität des Suchwerkzeuges zu verbessern. Die Hypothese dieses Ansatzes besteht darin, eine Suchanfrage automatisch generieren zu können, die dem

persönlichen Informationsbedürfnis mindestens so nahe kommt, wie eine vom Anwender direkt formulierte Suchanfrage. Dabei kann auch das persönliche Interessenprofil, welches über den aktuellen Kontext hinausgeht, eine Rolle spielen.

Im nächsten Abschnitt sollen Konzepte vorgestellt werden, durch deren Anwendung der textuelle Kontext der Aktivitäten des Anwenders auf eine Suchanfrage abgebildet werden kann. Zuvor sei jedoch auf ein Problem hingewiesen: die Heterogenität des Kontextes. Es kann nicht vorausgesetzt werden, dass ein Anwender ununterbrochen einem Interessengebiet nachgeht, so dass sich aus der Betrachtung eines bestimmten Zeitraumes (beginnend durch eine Initialaktion, etwa dem Starten eines Textverarbeitungsprogrammes) nicht zwangsläufig ein homogenes Informationsbedürfnis ableiten lässt.

Eine Möglichkeit, dieses Problem anzugehen, ist die der Bildung von Informations-Clustern. Diese Möglichkeit wird hier jedoch nicht weiter verfolgt.

In dieser Arbeit soll statt dessen der Kontext, der für die automatische Formulierung einer Suchanfrage betrachtet wird, auf die aktuell fokussierte Anwendung beschränkt werden. Diese Entscheidung basiert auf einer Vermutung, die sich auf Erfahrungswerte stützt. Es wird davon ausgegangen, dass in den meisten Anwendungsfällen, in denen ein Suchagent zum Einsatz kommt, die Informationen innerhalb der Anwendungen homogen sind, bezogen auf das Informationsbedürfnis des Anwenders. Beim Surfen in unterschiedlichen Rubriken eines Nachrichtenportals wird das Informationsbedürfnis demnach dennoch als homogen betrachtet („aktuelle Nachrichten“), auch wenn die Rubriken thematisch verschieden sind. Die Homogenität bleibt so lange bestehen, bis der Anwender sein

Informationsbedürfnis ändert. Dies soll an der Änderung des Anwendungsfokusses erkannt werden. Betrachtet werden dabei lediglich die Anwendungen, die vom Nutzer während dieses Anwendungsfalls fokussiert werden (d.h. Anwendungen, die durch einen Klick oder durch einen Taskswitch in den Vordergrund gebracht werden).

Es wurde nun definiert, *welcher* Text zur automatischen Generierung einer Suchanfrage herangezogen werden soll. *Wie* dies geschieht, ist eine Frage der Implementation. Dennoch muss man sich bereits in der Konzeptionsphase im Klaren darüber sein, dass es überhaupt technisch möglich ist, wenn eine Implementation möglich sein soll. Als eine Möglichkeit des Zugriffs auf andere Applikationen in 32-bit-Windows-Betriebssystemen bieten viele Programmiersprachen eine Win32API an.[11]

Im Folgenden soll eine Methode näher betrachtet werden, die es ermöglicht, den Text zu analysieren, von dem angenommen wird, dass er derzeit im Interesse des Anwenders steht. Analysieren heißt, eine Suchanfrage zu formulieren, die den Text informell repräsentiert.

Termfrequenz-Analyse

Ein *Lexem* ist ein Ausdruck der natürlichen Sprache, der innerhalb eines Textes von Wortbegrenzungszeichen (Leerzeichen, Satzzeichen) umgeben ist. Unter dem Begriff *Term* sei eine semantische Einheit innerhalb eines Textes verstanden, z.B. Eigennamen wie „Wolfgang Tiefensee“. Eigennamen sind bei einer Betrachtung auf Lexem-Ebene für die Informationssuche weniger geeignet als Terme. Eine Suche nach „Wolfgang“ + „Tiefensee“ ist erfahrungsgemäß weniger speziell als eine Suche nach „Wolfgang Tiefensee“.

[11] Als Stichwörter seien an dieser Stelle die Funktion ReadProcessMemory und die Message WM_COPY genannt.

Da gerade Eigennamen bei der Informationssuche eine große Rolle spielen [Friburger, 2002], wäre es wünschenswert, eine Segmentierung des Textes auf semantischer Ebene durchzuführen, d.h. auf Termebene. Dies erfordert einen hohen computerlinguistischen Aufwand[12]. In dieser Arbeit wird auf eine Segmentierung auf semantischer Ebene verzichtet.

Es gibt jedoch auch statistische Methoden, um semantisch zusammengehörige Lexeme aufzuspüren: Verfahren zur Extraktion von Kollokationen. Bei Eigennamen besteht eine Kollokation aus mehreren Lexemen, die in ihrer Gesamtheit einen Sinn ergeben, der sich nicht aus den einzelnen Lexemen ableiten lässt. So wird beispielsweise die Wortgruppe „ins Gras beißen“ als Kollokation bezeichnet (speziell: Idiom).

Statistische Verfahren zur Kollokationssuche machen es sich zunutze, dass die Lexeme einer Kollokation in Texten häufiger gemeinsam Auftreten als beliebige andere Lexeme gemeinsam auftreten. Untersucht man beispielsweise einen Text, der über die Leipziger Olympiabewerbung Bericht erstattet, so wird auffallen, dass die Lexeme *Wolfgang* und *Tiefensee* gemeinsam (direkt nebeneinander) häufiger auftreten, als die Lexeme *Wolfgang* und *Sport*. Sie können so als Kollokation erkannt werden.

$$p(k|a,b,N) = \frac{b!}{(b-k)!\,k!} \cdot \frac{a}{N} \cdot \frac{a-1}{N-1} \cdots \frac{a-k+1}{N-k+1} \cdot \frac{N-a}{N-k} \cdot \frac{N-a-1}{N-k-1} \cdots \frac{N-a-b+k+1}{N-b+1}$$

Abbildung 10: Wahrscheinlichkeit für gemeinsames Auftreten zweier Wörter in einem Satz

In Abbildung 10 soll p(k | a, b, N) berechnet werden, also die Wahrscheinlichkeit, dass die Wörter 1 und 2 k Mal gemeinsam in einem Satz auftauchen, gegeben dass das Wort 1 in a Sätzen und das Wort 2 in b Sätzen in einem Text mit N Sätzen auftauchen. Anhand eines Hypothesen-Tests kann

[12] Stichwörter: Satzsegmentierung, Part-of-Speech-Tagging, Markov Modell

eine Aussage über die Signifikanz der Kollokation (Wort 1, Wort 2) getroffen werden.

Nach der Segmentierung des Textes in Terme (Lexeme und Kollokationen) ist es möglich, Terme zu selektieren. Betrachten wir dies anhand eines konkreten Textes (aus dieser Arbeit):

> *Das Wort Information leitet sich von informatio ab, einem Wort, das in der lateinischen Umgangssprache gebraucht wurde. Die in dem Wort enthaltene Wurzel forma geht auf Begriffe der griechischen Philosophie zurück (eidos, idea, morphé, typos), die in diesem Zusammenhang von den bedeutenden Philosophen Platon und Aristoteles gebraucht wurden.*

Beispiel 5: Beispieltext zur Erläuterung der Termfrequenz-Analyse

Um den Text auf wenige Terme zu reduzieren (um damit eine Suchanfrage durchführen zu können), müssen die Terme nach ihrer inhaltlichen Relevanz für den Text beurteilt werden können. Die Relevanz gibt Auskunft darüber, wie stark ein Term den Text inhaltlich repräsentiert.

Bei manuellem Betrachten der ersten beiden Wörter fällt auf, dass der Term „Das“ kaum relevant sein dürfte, der Term „Wort“ in diesem Zusammenhang jedoch durchaus. Während wir dies manuell feststellen, fließt allerdings viel Wissen in die Betrachtung ein, wie kann eine solche Betrachtung automatisiert werden?

Eine Möglichkeit auf nicht-semantischer Ebene ist das Heranziehen der Frequenz von Termen. Stellt man den Text (Dokument *d*) als Vektor $\in IR^t$ dar, wobei jeder Term $t = |T|$ genau einer Dimension des Raumes und die Häufigkeit $h(t)$ den Ausprägungen entsprechen, so ergibt sich folgender Vektor:

$$\vec{d} = \begin{pmatrix} h(Das) \\ h(Wort) \\ h(Information) \\ ... \\ h(Aristoteles) \\ h(gebraucht) \\ h(wurde) \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \\ 1 \\ ... \\ 1 \\ 2 \\ 2 \end{pmatrix}$$

Abbildung 11: Termfrequenz-Vektor für den Text aus Beispiel 5 (Auszug)

Groß- und Kleinschreibung werden ignoriert. Für weitere Berechnungen ist es zunächst notwendig, den Vektor zu normieren. Dies erreichen wir, indem wir die Termhäufigkeit durch die relative Termhäufigkeit ersetzen, bezogen auf die Termanzahl:

$$\vec{d} = \frac{1}{t} \begin{pmatrix} h(Das) \\ h(Wort) \\ h(Information) \\ ... \\ h(Aristoteles) \\ h(gebraucht) \\ h(wurde) \end{pmatrix} = \begin{pmatrix} 2/47 \\ 3/47 \\ 1/47 \\ ... \\ 1/47 \\ 2/47 \\ 2/47 \end{pmatrix}$$

Abbildung 12: normierter Termfrequenz-Vektor für den Text aus Beispiel 5

Das Ziel besteht nach wie vor darin, die inhaltliche Relevanz der einzelnen Terme zu bestimmen; in der Vektordarstellung entspricht dies der Ausprägung der einzelnen Dimensionen. In unserem Beispielvektor repräsentiert die Ausprägung derzeit die relative Frequenz, die jedoch nicht zur Beschreibung der inhaltlichen Relevanz taugt. Deshalb benötigen wir einen weiteren Vektor: einen Vektor, der die Frequenz der im Dokument auftauchenden Terme in allgemeinen Texten wiedergibt. In Abbildung 12 wird deutlich: die Termfrequenz des Terms *Aristoteles* beträgt 1/47 und somit nur die Hälfte der

Termfrequenz des Terms *wurde*. Verglichen mit der Termfrequenz in allgemeinen Texten (Abbildung 13) zeigt sich, dass der Term *Aristoteles* im Verhältnis zum Term *wurde* im Beispieltext deutlich häufiger auftaucht, als in allgemeinen Texten.

$$\vec{a} = \begin{pmatrix} h_a(Das) \\ h_a(Wort) \\ h_a(Information) \\ \dots \\ h_a(Aristoteles) \\ h_a(gebraucht) \\ h_a(wurde) \end{pmatrix} = \begin{pmatrix} 0{,}008496 \\ 0{,}000107 \\ 0{,}000030 \\ \dots \\ 0{,}000002 \\ 0{,}000022 \\ 0{,}001561 \end{pmatrix}$$

Abbildung 13: Vektor für Termwahrscheinlichkeit in allgemeinen Texten bezüglich Text aus Beispiel 5

Der Termfrequenzvektor des Beispieltextes soll nun bezüglich der Termfrequenzen dieser Terme in allgemeinen Dokumenten relativiert werden, damit schließlich eine Aussage darüber getroffen werden kann, welche Terme den Beispieltext von allgemeinen Texten unterscheiden.

$$\vec{r} = \frac{1}{t}\begin{pmatrix} h(Das)/h_a(Das) \\ h(Wort)/h_a(Wort) \\ h(Information)/h_a(Information) \\ \dots \\ h(Aristoteles)/h_a(Aristoteles) \\ h(gebraucht)/h_a(gebraucht) \\ h(wurde)/h_a(wurde) \end{pmatrix} = \begin{pmatrix} 2/47/0{,}008496 \\ 3/47/0{,}000107 \\ 1/47/0{,}000030 \\ \dots \\ 1/47/0{,}000002 \\ 2/47/0{,}000022 \\ 2/47/0{,}001561 \end{pmatrix} = \begin{pmatrix} 5 \\ 597 \\ 709 \\ \dots \\ 10638 \\ 1934 \\ 27 \end{pmatrix}$$

Abbildung 14: Vektor für den Text aus Beispiel 5: relative Häufigkeit der Terme bezüglich allgemeiner Texte

Je höher die relative Häufigkeit eines Terms gegenüber allgemeiner Texte ist, als desto relevanter wird dieser Term für den konkreten Text eingestuft. Für die sechs Terme des Beispieltexts fällt das Ranking nach Relevanz folgendermaßen

aus: *Aristoteles*, *gebraucht*, *Information*, *Wort*, *wurde*, *Das*. Für eine Interpretation sollten mehr als sechs Ausdrücke des Beispieltextes betrachtet werden, es läßt sich dennoch feststellen: die Terme ohne lexikalische Bedeutung *Das* und *wurde* werden von dem Verfahren als am wenigsten relevant eingestuft. Wörter ohne lexikalische Bedeutung werden auch Funktionswörter genannt; *Das* und *wurde* zählen tatsächlich zu den Funktionswörtern. Der Term Aristoteles tritt 10638 Mal so häufig auf wie in allgemeinen Texten. Er wird damit als am meisten relevant für die betrachteten Wörter des Beispieltextes gewertet.

Anhand des Ranking kann nun eine bestimmte Anzahl von Termen selektiert werden, die schlagwortartig für den Gesamttext stehen. Im Information Retrieval wird dieser Vorgang auch als Feature Selection bezeichnet. Diese Terme können in eine Suchanfrage überführt werden. Es stellt sich die Frage, wie die Terme in der Suchanfrage verknüpft werden (UND / ODER) und wie viele Terme in einer Suchanfrage verwendet werden. Auf dieses Detail soll an dieser Stelle nicht konkret eingegangen werden, es sind jedoch folgende Vorgehensweisen denkbar:

Bevor eine Implementation vorgenommen werden kann, müssen empirische Untersuchungen zeigen, wie sich Termanzahl und Verknüpfung auf die Qualität der Suchanfrage auswirken. Die Ergebnisse werden als veränderbare Standardeinstellung in das Suchwerkzeug übernommen.

Ausgehend von den Standardeinstellungen wäre eine automatische Anpassung der Termanzahl und/oder Verknüpfung durch das Suchwerkzeug denkbar. So könnten etwa die Auswirkungen der Änderung der Termanzahl direkt nachvollzogen werden, indem die Suchanfrage mit unterschiedlicher Termanzahl automatisch mehrfach gestellt würde. Anhand der Suchergebnisse ließe sich der Zusammenhang evaluieren und bei geeigneten Lernmethoden somit eine dynamische Termanzahl realisieren.

Es wurde im letzten Kapitel mehrfach von *allgemeinen Texten* geredet. Es stellt sich natürlich die Frage: Was sind allgemeine Texte und wie kann das Suchwerkzeug den Vektor für Termwahrscheinlichkeit in allgemeinen Texten (siehe Abbildung 13) bestimmen bzw. aktuell halten?

Für das beschriebene Verfahren ist der optimale allgemeine Text die Summe aller Texte, über die die Suchanfrage ausgeführt wird, da dies im Rahmen der Suche der *Welt* entspricht, d.h. $|D|$, wenn das Dokument *d* betrachtet wird. Dies ist aus technischer Sicht jedoch problematisch: zum einen soll die Suchmaschine nicht festgelegt werden, es sogar möglich sein, eine Suchanfrage auf mehreren Suchmaschinen auszuführen und zum anderen ist anzunehmen, dass die Betreiber der Suchmaschinen ihren Datenbestand nicht zur Verfügung stellen.

Für das Beispiel wurde auf den Korpus des Wortschatzlexikons[13] zurückgegriffen, einem Projekt der Universität Leipzig, bei dem über 35 Mio. Sätze mit über 500 Mio. Wörtern untersucht wurden. Aufgrund des Umfangs und der vielfältigen und sorgfältig ausgewählten Quellen liegt die Vermutung nahe, dass sich der Korpus für diese Aufgabe eignet.

Doch auch bei einem solch umfangreichen Korpus ist die Frage der Erweiterung und Aktualisierung durchaus zu stellen. Gerade im Nachrichtenbereich tauchen immer wieder neue Eigennamen auf, die dem Korpus hinzugefügt werden müssen. Es bietet sich an, dass das Suchwerkzeug fortwährend für alle gefundenen Dokumente, d.h. Dokumente die von den Suchmaschinen vorgeschlagen wurden, eine Termanalyse vornimmt, um neue Terme dem Korpus hinzuzufügen. Dabei sollte eine Plausibilitätsprüfung zum Einsatz kommen, die verhindert, dass der Korpus mit Termen „verunreinigt" wird, die keine lexikalische Bedeutung haben.[14]

[13] www.wortschatz.uni-leipzig.de (Abgerufen am 11.7.2003)
[14] z.B. wenn versehentlich Binärdaten als Textdaten interpretiert werden; Stichwort zur kostengünstigen Plausibilitätsprüfung: n-Gram Analysen

Zum Beginn dieses Kapitels wurde bereits auf die Ähnlichkeit dieses Konzeptes mit den Konzepten der Interfaceagenten hingewiesen. Tatsächlich lassen sich Parallelen zwischen der Ermittlung des indirekten Informationsbedürfnisses (d.h. mit der automatischen Generierung einer Suchanfrage) und den Agentenkonzepten aufzeigen: Das Suchwerkzeug reagiert auf seine Umwelt, d.h. auf die Anwendungsumgebung des Benutzers. Von den Handlungen des Benutzers hängt es ab, ob das Suchwerkzeug automatisch eine Suchanfrage stellt oder nicht. So wird beispielsweise ein Informationsbedürfnis vermutet, wenn ein Textdokument oder eine Website nach dem Laden so lange fokussiert bleibt, dass davon ausgegangen werden kann, dass der Benutzer dieses Dokument liest. Dieses Verhalten ist fortgeschritten reaktiv.

In Abbildung 15 wird Phase 1 abschließend schematisch dargestellt. Wie zu sehen ist, wird die Suchanfrage, die entweder direkt vom Anwender eingegeben wurde oder indirekt aus den Handlungen des Anwenders abgeleitete wurde, an Phase 2 übergeben.

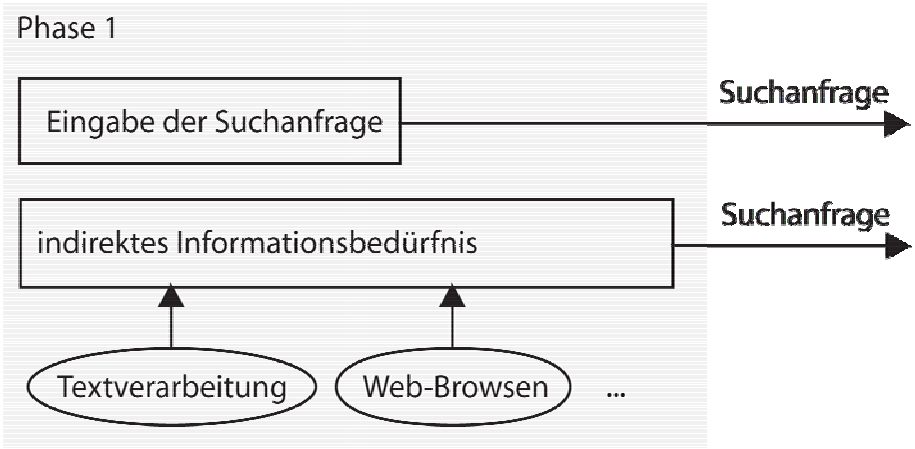

Abbildung 15: Phase 1 – Bestimmung des Informationswunsches; Unterscheidung zwischen direkter Eingabe der Suchanfrage und indirektem Informationsbedürfnis

3.3.2. Phase 2 – Informationsgewinnung und Personalisierung

In Phase 1 wurde eine Suchanfrage ermittelt, die das Informationsbedürfnis des Anwenders beschreibt. Diese Suchanfrage hat bereits ein Format, welches von den herkömmlichen Suchmaschinen verarbeitet werden kann. Um eine individuelle Informationssuche zu ermöglichen wird die Suchanfrage jedoch zunächst personalisiert. Danach soll die Schnittstelle zu den Suchmaschinen näher betrachtet werden.

In Abbildung 16 werden diese zwei wesentlichen Aufgaben der Phase 2 veranschaulicht: Als Eingabe dient die in Phase 1 ermittelte Suchanfrage. Diese wird zunächst durch das Personalisierungsmodul anhand des persönlichen Nutzerprofils des Anwenders individualisiert. Dabei wird auch auf das Nutzerprofil Einfluss genommen, da die Suchanfrage Ausdruck der Bedürfnisse des Nutzers ist. Die personalisierte Suchanfrage wird an das Informationsgewinnungsmodul weitergeleitet. Anhand des Wissens über die Interfaces der Suchmaschinen greift dieses auf die Datenbestände der Suchmaschinen zu und ermittelt die Verweise auf die Zieldokumente aus den HTML-Ergebnisseiten der Suchmaschinen. Die Zieldokumente werden im Rahmen der Informationsgewinnung geladen und schließlich an Phase 3 übergeben.

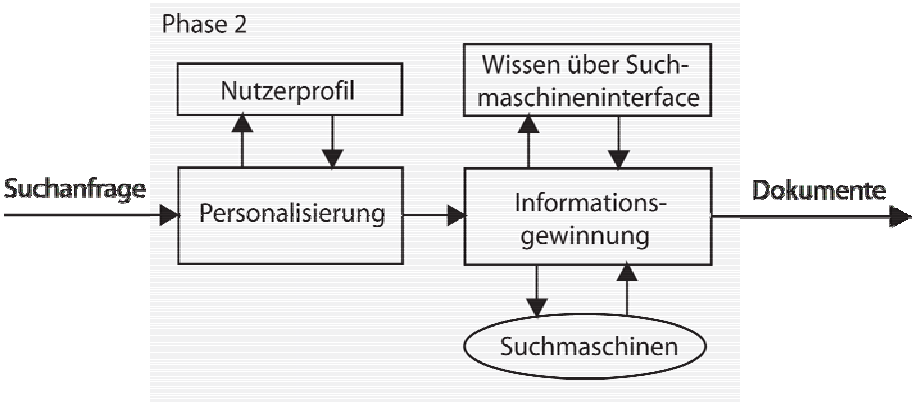

Abbildung 16: Phase 2 – Personalisierung und Informationsgewinnung; Übergang von der Suchanfrage zu den Ergebnisdokumenten

3.3.2.1. Personalisierung

Eine Stärke gegenüber herkömmlicher Suchmaschinen aufgrund der Clientseitigen Ausführung ist die Tatsache, dass der Anwender des Suchwerkzeuges als eine Person betrachtet werden kann. Damit ist es möglich, ein persönliches Profil dieser Person zu ermitteln, ein Nutzerprofil, das es ermöglicht, die Informationssuche individuell anzupassen. Die Hypothese, die zum Einsatz der Personalisierung führt ist die, dass damit die Qualität der Suchergebnisse verbessert werden kann.

In Phase 2 wird zunächst das Nutzerprofil mittels der Suchanfrage beeinflusst. Da jede Suchanfrage ein Informationsbedürfnis des Nutzers darstellt, kann diese direkt in das Nutzerprofil einfließen. Es ist jedoch zu beachten, dass nicht jedes Informationsbedürfnis über einen längeren Zeitraum besteht. Deshalb sind Methoden notwendig, die den natürlichen Vorgang des „Vergessens“ auf das Nutzerprofil abbilden.

Ein Beispiel soll zeigen, was unter „Vergessen“ verstanden werden soll: Sucht ein Nutzer erstmalig nach dem Term „Platon“, so kann dies als temporäres Informationsbedürfnis betrachtet werden. Der Nutzer ist möglicherweise nur in der aktuellen Situation an Informationen zu diesem Term interessiert. Nach einem bestimmten Zeitraum, der „Merkdauer“, soll der Term keinen weiteren Einfluss auf das Nutzerprofil nehmen. Wird der Term jedoch erneut gesucht, dann muss berücksichtigt werden, dass bereits nach diesem Term gesucht wurde, auch wenn er zwischenzeitlich „in Vergessenheit geraten“ ist. Umgangssprachlich kommt dies dem „in Erinnerung rufen“ nahe.

Innerhalb des Suchwerkzeuges soll das Nutzerprofil ausschließlich inhaltliche Interessen des Nutzers repräsentieren. Da Inhalte textuell repräsentiert werden, d.h. durch Terme, bietet sich das Term-Vektor Modell aus dem Abschnitt Termfrequenz-Analyse an (Seite 3-69): als Terme werden die einzelnen Begriffe der Suchanfragen in den Vektor aufgenommen. Um die

Betrachtung von Zeiträumen zuzulassen, kann die Ausprägung dieser Terme jedoch nicht einfach der Häufigkeit oder Termfrequenz entsprechen. Es sind vielmehr Informationen über das Auftreten dieses Terms in der Vergangenheit notwendig, um zu beurteilen, ob der Term „in Vergessenheit geraten“ soll, oder nicht. In Abbildung 17 wird das Auftreten von Termen (y-Achse) über die Suchanfragen (x-Achse) dargestellt.

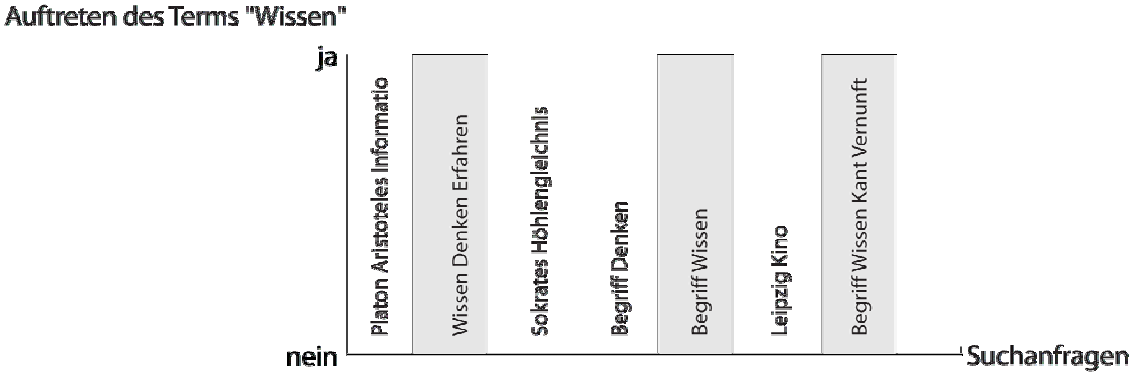

Abbildung 17: Binäre Abbildung eines Terms von Suchanfragen; geeignet für das Term-Vektor-Modell

Wie zu sehen ist, wird nur zwischen Auftreten und Nicht-Auftreten eines Terms unterschieden, die Auflösung der y-Achse ist also binär. Pro Suchanfrage wird so je Term ein Bit benötigt. Mit dieser Datenstruktur lässt sich effektiv eine große Anzahl von Suchanfragen speichern und später rekonstruieren. Zeitliche Abläufe werden nur insofern berücksichtigt, als das die Reihenfolge der Suchanfragen festgehalten wird. Damit die zu Speichernde und zu Verarbeitende Datenmenge nicht zu groß wird, kann eine maximal zu „merkende“ Anzahl von Suchanfragen festgelegt werden. Die in Abbildung 17 gezeigte Bitfolge entspräche der Ausprägung im Term-Vektor-Modell mittels dessen die Suchanfragen im Nutzerprofil gespeichert werden. Wird eine neue Suchanfrage ausgeführt, so werden die entsprechenden Bits angefügt, bzw. ggf. der bisher nicht vorhandene Term in den Vektor aufgenommen.

Nachdem gezeigt wurde, wie das Nutzerprofil entsteht und erweitert wird, soll betrachtet werden, wie dies in die Suchanfragen einfließen kann. Dabei wird zunächst unterschieden zwischen direkt formulierter Suchanfrage und indirekt automatisch ermittelter Suchanfrage. Um dem Anwender eine volle Kontrolle über die Suchanfragen zu ermöglichen, sollen nur automatisch ermittelte Suchanfragen verändert werden. Anfragen, die er direkt eingibt, werden unverändert ausgeführt.

Warum soll die Suchanfrage überhaupt nachträglich verändert werden? Es ist davon auszugehen, dass die in Phase 1 automatisch ermittelte Suchanfrage das Informationsbedürfnis des Nutzers weniger gut beschreibt, als hätte der Nutzer diese Suchanfrage selbständig formuliert. Deshalb soll das Wissen über die bisherigen Suchanfragen genutzt werden, um die aktuelle Anfrage zu konkretisieren. Die Vorgehensweise soll an einem Beispiel erläutert werden:

Der Nutzer liest einen Text über den Begriff „Information“ (Beispiel 5). In Phase 1 wird während des Lesens die automatische Suchanfrage „Aristoteles gebraucht Information“ ermittelt (Abbildung 14). Vom Personalisierungsmodul wird diese Suchanfrage mit dem Nutzerprofil verglichen. Dazu werden die Suchanfragen des Nutzerprofils als Cluster betrachtet und das Cluster mit der größten Ähnlichkeit gesucht. Der Ähnlichkeitsbegriff wird zunächst auf eine Übereinstimmung von Termen beschränkt, es sind jedoch auch Fuzzy-Vergleiche denkbar, die einen mächtigeren Vergleich zulassen. Es wird festgestellt, dass bereits eine ähnliche Suchanfrage gestellt wurde: „Platon Aristoteles Information“. Da diese Suchanfrage der aktuellen Anfrage zu 2/3 ähnelt, eine Mindestähnlichkeit von 50% demnach gegeben ist[15], wird angenommen, dass auch der nicht ähnliche Teil der Suchanfrage zur

[15] Das Mindestähnlichkeitsmaß wurde hier aus Gründen der Verständlichkeit willkürlich festgelegt. Die optimale Einstellung dieses Parameters muss anhand von Test bestimmt werden.

Konkretisierung der Anfrage genutzt werden kann. Im Beispiel würde der Term „Platon“ zur automatischen Suchanfrage „Aristoteles gebraucht Information“ hinzugefügt. Die Hypothese dieses Verfahrens ist folgende:

Automatische Suchanfragen sollten konkret statt allgemein sein. Allgemeine Dokumente würden den Nutzer von seinem aktuellen Interesse ablenken. Der Gefahr, dass eine automatische Konkretisierung nicht das jeweils konkrete Interesse des Nutzers trifft, kann in Phase 3 dadurch entgegengewirkt werden, dass die Ergebnisdokumente der automatischen Suche mit dem Ausgangstext inhaltlich verglichen werden.

Es wird vermutet, dass manuelle Suchanfragen das Informationsbedürfnis besser repräsentieren als automatische. Deshalb werden die automatischen Suchanfragen anhand der bisher manuell getätigten Suchanfragen konkretisiert. Dies erfordert die Speicherung der Information, ob eine Suchanfrage manuell oder automatisch erzeugt wurde. Es bietet sich an, im Term-Vektor des Nutzerprofils eine weitere Dimension mit binärer Ausprägung hinzuzufügen. Ein gesetztes Bit würde in der entsprechenden Suchanfrage so beispielsweise einer automatischen Anfrage entsprechen, während ein gelöschtes Bit eine manuelle Suche kennzeichnet.

Dieses Personalisierungsverfahren ist umso Erfolg versprechender, je mehr Suchanfragen im Nutzerprofil vorliegen.

3.3.2.2. Informationsgewinnung

Wie bereits erwähnt wurde, ist es erstrebenswert, das Suchwerkzeug nicht auf eine Suchmaschine festzulegen. Da das Suchwerkzeug auf Client-Seite ausgeführt wird, entzieht sich die Pflege der Schnittstelle der direkten Kontrolle der Entwickler; die Kontrolle über die Schnittstelle der Suchmaschine obliegt ohnehin ausschließlich den Suchmaschinenbetreibern. All diese Punkte machen deutlich, dass eine Flexibilität beim Zugriff auf Suchmaschinen durch das

Suchwerkzeug wünschenswert wäre. Diese Flexibilität sollte sowohl eine automatische Anpassung bei Änderungen im Suchmaschineninterface ermöglichen als auch den Zugriff auf gänzlich neue Suchmaschinen zulassen.

Nachdem die Anforderungen an das Informationsgewinnungsmodul bestimmt wurden, soll die technische Vorgehensweise betrachtet werden.

Bestimmung des Suchmaschinen-Eingabeinterfaces

Die Eingabe einer Suchanfrage erfolgt bei den meistverbreiteten Suchmaschinen direkt auf der Startseite. Aus Performancegründen und Gründen der Usability ist es auch nicht zu erwarten, dass sich dies ändert. Durch eine Analyse des HTML-Dokumentes, welches vom Suchmaschinenserver auf die Anforderung der Start-URL ausgegeben wird, kann somit die Eingabeschnittstelle bestimmt werden.

Betrachtet man die Startseiten der meistverbreiteten Suchmaschinen, so fällt auf, dass die Suchanfrage immer über HTML-Input-Felder übermittelt wird. Im Fall der Suchmaschinen unter www.google.de und www.aol.de konnte genau ein HTML-Input-Feld ermittelt werden. Das Suchwerkzeug hat demnach nur nach dem entsprechenden HTML-Quelltextmuster zu suchen, um das Interface zu analysieren: Es ist zunächst der Quelltextbereich zwischen `<form ...>` und `</form>` zu extrahieren. Innerhalb dieses Bereiches ist nach einem `<input ...>` Element zu suchen. Aus der Zieladresse des `<form ...>` Elementes und dem Namen des `<input ...>` Elementes lässt sich die URL zur Übermittlung der Suchanfrage konstruieren.

Im Beispielfall www.msn.de wird jedoch deutlich, dass die Startseite auch mehrere Formularfelder enthalten kann. Hier ist zur Bestimmung des Suchfeldes eine Heuristik notwendig, die auf folgenden Kriterien basieren kann:

- längere Eingabefelder: Es wird angenommen, dass das Hauptsuchfeld länger ist als Suchfelder, die nicht der eigentlichen Suche in der Suchmaschine dienen (z.B. Suche nach Aktienkursen, Suchfelder in Werbeanzeigen etc.). Die Länge kann aus dem `size` Attribut des jeweiligen `<input ...>` Elementes entnommen werden.
- „search"/„suche": Das Wort „search" bzw. „suche" taucht innerhalb der Hauptsuchfelder erfahrungsgemäß oftmals in der URL, dem Formularnamen, den Elementnamen oder Element-IDs oder im `value` Attribut des Submit-Buttons auf.
- Position: Da in westlichen Kulturen von links oben nach rechts unten gelesen wird, kann davon ausgegangen werden, dass Hauptsuchfelder aus Gründen der Usability eher links oben positioniert werden.

Bestimmung des Suchmaschinen-Ausgabeinterfaces

Wurde ermittelt, wie die Suchanfragen an die Suchmaschine übergeben werden können, so müssen auch Verfahren zum Einsatz kommen, die die Ausgabeseite der Suchmaschine so verarbeiten, dass die Ergebnislinks aus dem HTML-Quelltext separiert werden können. Die Analyse der Suchergebnisseiten ist bei den herkömmlichen Suchmaschinen komplexer als die Analyse der Startseiten. Auch hier kommt eine Heuristik zum Einsatz:[16]

- Linkziel: Viele Links auf den Ausgabeseiten verweisen nicht auf die Suchergebnisse, sondern auf weitere Funktionalitäten der Suchmaschine. So ist bei Google per Klick auf „Groups" die selbe Suche auch in den Newsgroups ausführbar. Interne Links können anhand der selben Domain innerhalb des Verweiszieles erkannt werden.

[16] Bei der Beschreibung der Heuristik werden Wörter wie „oft", „meist" und „unmittelbar" ohne nähere Erläuterung verwendet. Es sei darauf hingewiesen, dass bei der Implementation die Parameter der Heuristik anhand von Tests aufeinander abgestimmt werden müssen. Es soll in der Konzeptionsphase nicht weiter darauf eingegangen werden.

- Linkposition: Die Suchergebnisseiten der meistgenutzten Suchmaschinen haben folgenden Aufbau: Kopf mit Suchfeld, Werbung und weiteren Funktionalitäten der Suchmaschine; Körper mit Links zu den Ergebnisseiten; Fuß mit Suchfeld, Werbung und weiteren Funktionalitäten der Suchmaschine. Oft folgen die Suchergebnislinks innerhalb des Körpers einander.
- Suchbegriffe: Begriffe aus der Suchanfrage sind meist innerhalb des Suchergebnislinks oder im unmittelbar folgenden Text vorhanden.
- Struktur: HTML-Strukturierungsmöglichkeiten wie Listen können ein Indikator für die Ergebnislisten sein.

Nach der Ermittlung der Verweise zu den Zieldokumenten können diese Dokumente geladen werden. Dabei muss auch das Dokumentenformat berücksichtigt werden: es sollen nur Dokumente geladen werden, die in Phase 3 verarbeitet werden können. Im Rahmen dieses Konzeptes werden aus Gründen der Übersicht in Phase 3 nur HTML-Dokumente akzeptiert. Das Laden der Dokumente bedeutet natürlich einen zeitlichen Mehraufwand gegenüber den herkömmlichen Suchmaschinen. Wie bei den Anforderungen in Kapitel 3-57 in punkto Schnelligkeit festgestellt wurde, muss sich ein Suchwerkzeug an den herkömmlichen Suchmaschinen messen lassen. Es ist daher notwendig, dass die Links zu den Ergebnisdokumenten bereits vor dem Laden, und somit vor der Analyse, im Interface des Suchwerkzeuges erscheinen (ggf. mit einem grafischen Hinweis darauf, dass diese noch nicht analysiert wurden). Während Phase 3 wird diese Liste von Links sukzessive überarbeitet, d.h. sowohl das Ranking anhand der Analyseergebnisse korrigiert als auch Links entfernt, die für nicht relevant gehalten werden.

Während der Benutzer bei einer direkten Suchanfrage eine Suchdauer erwartet, die durch seine Erfahrungen mit den herkömmliche Suchmaschinen

geprägt ist und kein Laden der Zieldokumente zulässt, so ist die Situation bei einer indirekten Suchanfrage eine andere. Hier kann das Suchwerkzeug seine Stärke des Automatismus ausspielen. Da der Nutzer in seiner Arbeit (z.B. Lesen eines Textes) nicht unterbrochen wird, ist die Dauer des Suchvorgangs weniger von Belang.

3.3.3. Phase 3 – Informationsausgabe und Lernmethoden

In der dritten Phase werden die Suchergebnisse ausgegeben und Lernmethoden vollzogen, die abermals auf das Nutzerprofil Einfluss nehmen. In Abbildung 18 sind drei Module zu sehen: die Dokumentanalyse, Filterung und Ranking sowie die Lernmethoden. Zusammenspiel und Funktionsweise dieser Module sollen im Folgenden erläutert werden.

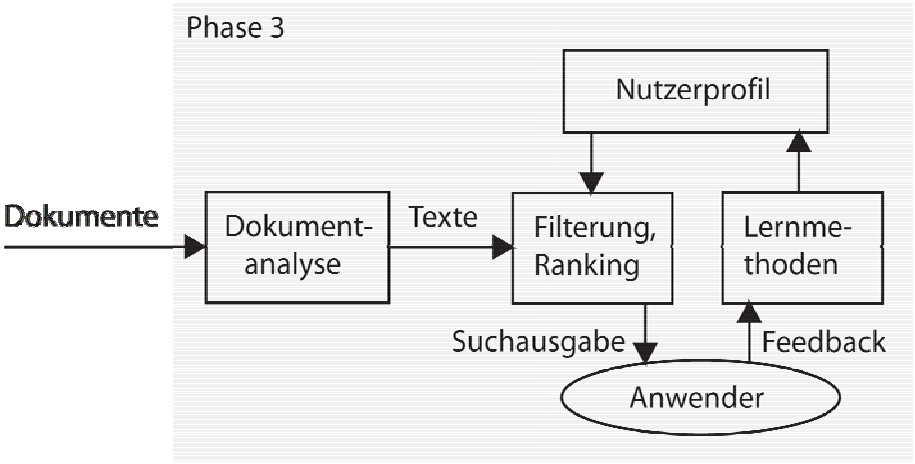

Abbildung 18: Phase 3 – Informationsausgabe und Lernmethoden; Auswertung und Ranking der Suchergebnisse; Lernmethoden anhand des Anwenderfeedbacks

3.3.3.1. Dokumentanalyse

Bei den an Phase 3 übergebenen Dokumenten handelt es sich um strukturierte Daten. Aus Gründen der Übersichtlichkeit dieser Arbeit wurde bereits einschränkend festgelegt, dass es sich nur um Dokumente im HTML-Format handelt. Erfahrungsgemäß ist damit die überwiegende Menge der über

herkömmliche Suchmaschinen erreichbaren Internetdokumente eingeschlossen. Das Ziel der Dokumentanalyse besteht darin, HTML-Dokumente in das Textformat umzuwandeln und dabei die Bereiche des HTML-Dokumentes herauszufiltern, die keine inhaltliche Funktion im Dokument erfüllen (Menüleisten etc.). Dieser Prozess dient der Vorbereitung der Daten für das nächste Modul: dem Ranking und Filterung der Dokumente. In Abbildung 19 wurden alle Bereiche einer Nachrichtenseite ausgegraut, die in die inhaltliche Weiterverarbeitung des Dokuments nicht einfließen sollen. Dazu gehören vor allem Elemente der Navigation.

Abbildung 19: inhaltlich relevanter Bereich einer Nachrichtenseite für die Dokumentanalyse; Menüs und sonstige herauszufilternde Informationen sind ausgegraut

Dieser Vorgang kann mit einer einfachen aber effektiven Heuristik vorgenommen werden:

- Funktionswörter: In inhaltlich nicht relevanten Bereichen werden selten Funktionswörter (Artikel, Pronomen, Präpositionen etc.) verwendet. Taucht in einem Abschnitt eine Mindestanzahl von Funktionswörtern nicht auf, dann kann dieser Abschnitt als Nicht-Inhaltsblock gewertet werden.
- Satzform: Inhaltlich relevante Bereiche können oft an Sätzen erkannt werden: beginnend mit Großbuchstaben und endend mit einem Satzzeichen.
- Position: Inhaltlich relevante Bereiche liegen oft direkt beieinander. So tritt Punkt 1 und Punkt 2 für Überschriften oder Zwischenüberschriften über einem Inhaltsblock oder in einem Inhaltsblock außer Kraft.

HTML-Strukturelemente, wie Tags zur Auszeichnung von Überschriften (z.B. `<h1>`) oder von Textblöcken (`<p>`), können bei der Analyse nicht berücksichtigt werden, da diese nur selten zur inhaltlichen Beschreibung eines Dokumentes genutzt werden, oft jedoch für gestalterische Zwecke.

3.3.3.2. Filterung und Ranking

Bei herkömmlichen Suchmaschinen liegt das Informationsbedürfnis des Nutzers ausschließlich in Form eine Suchanfrage vor. Anhand dieser Anfrage wird im Index der Suchmaschine nach passenden Dokumenten gesucht. Wurden Dokumente gefunden, so ist eine genauere Betrachtung der Dokumente nicht sinnvoll, weil bereits alle Kriterien berücksichtigt wurden, die über das Informationsbedürfnis des Nutzers vorliegen: die Terme der Suchanfrage.

Im hier konzipierten Suchwerkzeug liegen weitere Informationen über das Informationsbedürfnis des Nutzers vor: das Nutzerprofil und im Fall der automatisch generierten Suchanfrage der Ausgangstext, anhand dessen die automatische Suchanfrage erstellt wurde. Zwar war es notwendig, diese Informationen auf eine Suchanfrage zu reduzieren, um auf herkömmliche Suchmaschinen zuzugreifen, aber durch eine Analyse der Ergebnisdokumente können nun auch das Nutzerprofil und der Ausgangstext für eine Bewertung der Relevanz der Ergebnisdokumente herangezogen werden. Die Vermutung besteht darin, dass damit ein fortgeschritteneres Ranking erzielt werden kann, als dies bei herkömmlichen Suchmaschinen der Fall ist. Die Filterung besteht darin, dass irrelevante Dokumente gar nicht erst in die Ergebnisliste aufgenommen werden. Dies sind Dokumente, die zwar aufgrund der Relevanz für die Suchanfrage von den herkömmlichen Suchmaschinen gefunden wurden, jedoch bei einem nachträglichen Vergleich mit dem Ausgangsdokument und dem Nutzerprofil als nicht relevant eingestuft werden. Außerdem werden automatisch Dokumente gefiltert, die veraltet oder verschwunden sind und somit auf einen veralteten Suchmaschinenindex zurückzuführen sind.

Filterung und Ranking sind als Bestimmung eines Ähnlichkeitsmaßes zwischen den Ergebnisdokumenten und der Suchanfrage, dem Nutzerprofil und dem Ausgangstext (so vorhanden) zu betrachten, wobei die Filterung nur einem Sonderfall entspricht, nämlich dem der Unterschreitung eines Mindestähnlichkeitsmaßes. Es soll im Folgenden ein Verfahren zur Ermittlung eines Ähnlichkeitsmaßes vorgestellt werden.

Ähnlichkeitsmaß

Mit den Ergebnisdokumenten kann so vorgegangen werden wie mit einem zu analysierenden Text im Abschnitt Termfrequenzanalyse (ab Seite 3-69). Das

Ergebnisdokument kann somit als Term-Vektor dargestellt werden. Da sowohl Suchanfrage und Nutzerprofil als auch Ausgangstext (so vorhanden) in Term-Vektor-Form vorliegen, liegt es nahe, das Ähnlichkeitsmaß auf Basis der Ähnlichkeit zweier Vektoren zu bestimmen.[17]

Es soll jedoch zunächst betrachtet werden, welche Vektoren in den konkreten Anwendungsfällen herangezogen werden. Es wird unterschieden zwischen dem direkten Informationsbedürfnis (Formulierung einer Suchanfrage) und der automatischen Generierung einer Suchanfrage. Wird vom Nutzer die Suchanfrage direkt formuliert, so liegt kein Ausgangstext vor, der das Informationsbedürfnis beschreibt. Das Ähnlichkeitsmaß kann nur anhand der Suchanfrage und anhand des Nutzerprofils bestimmt werden.

Wurde die Suchanfrage automatisch generiert, so geschah dies anhand eines Ausgangstextes. Dieser wurde nur aus dem Grund auf einige wenige Terme reduziert, damit mit Hilfe dieser auf Suchmaschinen zugegriffen werden konnte. Zur Bestimmung des Ähnlichkeitsmaßes kann der Ausgangstext, neben dem Nutzerprofil, jedoch wieder verwendet werden. Die Suchanfrage bleibt hingegen unbeachtet, da diese nur eine inhaltlich annähernde Repräsentation des Ausgangstextes darstellt.

Für die Berechnung des Ähnlichkeitsmaßes ergeben sich keine Unterschiede, es muss lediglich bei der Auswahl der zu vergleichenden Term-Vektoren der Anwendungsfall beachtet werden.

Die Berechnung erfolgt in zwei Stufen: 1. der Berechnung der Ähnlichkeit zwischen Ausgangstext/Suchanfrage und Ergebnisdokument und 2. der

[17] Die Ausprägung des Nutzerprofil-Term-Vektors spiegelt die Verwendung der Terme in Suchanfragen wider. Damit dieser Vektor mit den anderen Term-Vektoren verglichen werden kann, muss der Nutzerprofil-Term-Vektor zunächst in einen äquivalenten Term-Vektor überführt werden. Dazu muss aus der Bitfolge des jeweiligen Terms unter Berücksichtigung des *Vergessens* ein Skalar berechnet werden, das die Relevanz dieses Terms für den Nutzer darstellt.

Berechnung der Ähnlichkeit zwischen Nutzerprofil und Ergebnisdokument (soweit ein Nutzerprofil vorhanden ist[18]).

Beide Stufen basieren auf dem Vergleich zweier Term-Vektoren. Als Ähnlichkeitsmaß für Vektoren wird der Winkel verwendet.

$$\ddot{A}hnlichkeit(\vec{x}_1, \vec{x}_2) = \cos\alpha = \frac{\vec{x}_1 \vec{x}_2}{|\vec{x}_1||\vec{x}_2|}$$

Formel 1: Cosinus-Ähnlichkeitsmaß für zwei Vektoren $\vec{x}_1, \vec{x}_2$

In Formel 1 wird das Ähnlichkeitsmaß anhand des Cosinus des Winkels α der Vektoren $\vec{x}_1, \vec{x}_2$ formuliert. Für normalisierte Vektoren lässt sich die Cosinus-Ähnlichkeit als Skalarprodukt der Vektoren schreiben:

$$\ddot{A}hnlichkeit(\vec{x}_1, \vec{x}_2) = \vec{x}_1 \vec{x}_2 = \sum_{j=1}^{n} x_{1j} x_{2j}$$

Formel 2: Cosinus-Ähnlichkeitsmaß für zwei normalisierte Vektoren $\vec{x}_1, \vec{x}_2$

Mit Formel 2 wird je nach Anwendungsfall die Ähnlichkeit zwischen Ausgangstext und Ergebnisdokument bzw. Suchanfrage und Ergebnisdokument berechnet, nachdem die Vektoren auf die gleiche Länge gebracht wurden (durch Hinzufügen der jeweils fehlenden Terme mit einer Ausprägung von 0). Mittels Formel 2 wird auch die Ähnlichkeit zwischen Nutzerprofil und Ergebnisdokument berechnet. Das Gesamtähnlichkeitsmaß ergibt sich aus dem Mittel der beiden Ähnlichkeiten, wobei diese unterschiedlich gewichtet werden: das Nutzerprofil soll um ein vielfaches schwächer in das Gesamtähnlichkeitsmaß einfließen als das Ausgangsdokument bzw. die Suchanfrage. Die konkrete Gewichtung muss anhand von Tests optimiert werden.

[18] Ein Mindestumfang des Nutzerprofils sollte für diese Operation vorausgesetzt werden, um anfängliche einseitige Einflüsse aufgrund von Datenmangel zu vermeiden. Aus anwendungspolitischer Sicht ist es außerdem sinnvoll, das Abschalten der Personalisierungsfunktionen (und somit des Nutzerprofils) zu ermöglichen.

Das Ähnlichkeitsmaß wird für jedes Dokument ermittelt. Wie zum Beginn des Kapitels erwähnt, soll ein Mindestähnlichkeitsmaß als Filter für irrelevante Dokumente dienen. Auch dieser Schwellenwert muss anhand von Tests ermittelt und optimiert werden. Anhand der Ähnlichkeiten kann nun ein Ranking der Dokumente vorgenommen werden. Dabei spielt es keine Rolle, von welcher Suchmaschine das Dokument gefunden wurde, es kann also eine suchmaschinenübergreifende Liste von Ergebnisdokumenten sortiert nach der Relevanz ausgegeben werden.

3.3.3.3. Lernmethoden

Wie bei herkömmlichen Suchmaschinen wird auch im Interface des hier konzipierten Suchwerkzeuges eine Liste von Links ausgegeben, die zu den Ergebnisdokumenten führt. Da die Dokumente bereits für die Analyse geladen wurden, sind diese übrigens sofort verfügbar.

Durch das Auswählen eines dieser Links trifft der Nutzer eine Entscheidung auf der Basis seines Informationsbedürfnisses und der Erscheinung des Links (der Link-Überschrift und dem darunter erscheinenden Textauszug aus dem Dokument). Diese Entscheidung kann zunächst als Offenbarung des Informationsbedürfnisses an dem hinter dem Linktext vermuteten Inhalten gewertet werden. Es liegt nahe, dass die Terme, aus denen der Linktext besteht, in das Nutzerprofil aufgenommen werden. Dies geschieht mittels des selben Verfahrens wie bei der Personalisierung (ab Seite 3-78). Funktionswörter werden nicht berücksichtigt, da es um den inhaltlichen Charakter der Terme geht, der bei Funktionswörtern ignoriert werden kann.

Eine weitere Lernmethode berücksichtigt die Gegebenheit, dass beim Ranking eine Wichtung zwischen der Ähnlichkeit des Ergebnisdokuments und dem Nutzerprofil sowie zwischen Ausgangsdokument/Suchanfrage und

Ergebnisdokument vorgenommen wird. Dieser Parameter wird manuell initiiert, kann aber vom Suchwerkzeug zur Qualitätssteigerung während der Benutzung automatisch optimiert werden. Für die Bestimmung der Qualität des Parameters wird an dieser Stelle folgende Hypothese verwendet:

Kommt es signifikant häufig dazu, dass der Nutzer in der gerankten Ergebnisliste ein Dokument zuerst anklickt, welches weiter unten in der Liste (und somit im Ranking) steht, dann wird angenommen, dass 1) das Ranking mangelhaft ist oder 2) die Repräsentation des Dokumentes durch den Linktext mangelhaft ist. Zunächst wird 1) vermutet. Dazu vergleicht das Suchwerkzeug die Ähnlichkeiten der Term-Vektoren einzeln: die Ähnlichkeit zwischen Nutzerprofil und Ergebnisdokument und die Ähnlichkeit zwischen Ausgangsdokument/Suchanfrage und Ergebnisdokument. Es wird angenommen, dass die Gewichtung zwischen Nutzerprofil und Ausgangsdokument/Suchanfrage in der Berechnung der Gesamtähnlichkeit zu Gunsten des Nutzerprofils korrigiert werden muss, wenn das Nutzerprofil eine höhere Ähnlichkeit aufweist. Umgekehrt gilt: weisen Ausgangsdokument/Suchanfrage eine höhere Ähnlichkeit auf, dann wird die Gewichtung zu Gunsten dieser Komponente korrigiert. Kommt es trotz wiederholter Korrektur dazu, dass ein niedriger geranktes Dokument zuerst angeklickt wird, dann wird 2) als Grund dafür herangezogen. Auf eine dynamische Änderung des Linktextes soll hier allerdings nicht eingegangen werden.

4. Zusammenfassung und Ausblick

In dieser Arbeit wurde meist der Begriff *Suchwerkzeug* verwendet, während im Titel von einem *Suchagenten* die Rede ist. Nachdem im Kapitel Softwareagent (ab Seite 2-13) ausführlich auf Softwareagenten eingegangen wurde, soll nun abschließend geprüft werden, ob es gelungen ist, eine Software zu konzipieren, die dem Agentenbegriff gerecht wird. Dies geschieht anhand der Eigenschaften nach [Wooldridge, Jennings, 1995a]:

- **Autonomie:** Das Suchwerkzeug ist in der Lage, mit einem gewissen Maß an Eigenständigkeit einer Aufgabe nachzugehen. Insbesondere die automatischen Linkvorschläge, passend zu den Texten, die der Anwender aktuell liest bzw. schreibt, können als autonomes Handeln im Sinne eines Agenten betrachtet werden.

- **Soziale Fähigkeiten:** Das Konzept sieht keine Kommunikation mit anderen Agenten vor. Dennoch kann von sozialen Fähigkeiten geredet werden: das Suchwerkzeug kommuniziert mit dem Anwender nicht nur auf Basis der Suchanfragen. Durch die Einbindung des Nutzerprofils und der Lernmethoden werden auch indirekte und nonverbale Kommunikationsformen (z.B. Feedback durch Link-Klick) berücksichtigt und fließen in die Funktionalität ein.

- **Reaktivität/Proaktivität:** Die Umwelt des Suchwerkzeuges wird u. a. bestimmt durch sein Sprachverständnis (Informationen über Funktionswörter, Termfrequenzen in allgemeinen Texten, Segmentierungsverfahren etc.) und dem konkreten Informationsbedürfnis. Ändert sich das Informationsbedürfnis des Anwenders, so reagiert das Suchwerkzeug auf diese Änderung. Dies kann als einfaches reaktives Verhalten bezeichnet werden. Als

fortgeschritten reaktives oder sogar proaktives Verhalten können die Lernmethoden angeführt werden, anhand derer das Suchwerkzeug die Qualität des Rankings selbständig verbessert (ab Seite 3-91). Auch die bereits bei der Autonomie erwähnten automatischen Linkvorschläge können als fortgeschritten reaktives oder sogar proaktives Verhalten klassifiziert werden[19].

Damit können dem Suchwerkzeug die Eigenschaften eines Agenten bescheinigt werden. Der Begriff Suchagent ist daher gerechtfertigt.

Abschließend stellt sich die Frage: Warum haben sich Suchagenten nicht bereits stark verbreitet?

Es besteht die Vermutung, dass die Hemmschwelle zur Installation eines clientseitigen Suchwerkzeuges hoch ist, da sich serverseitige Suchmaschinen etabliert haben. Ein weiterer Grund kann die mangelnde Qualität der hier konzeptionell vorgestellten Funktionen bei einer konkreten Implementation sein.

Dennoch besteht Hoffnung, dass sich die Situation ändert. Möglicherweise spielen clientseitige Metasuchmaschinen[20], die inzwischen durchaus Akzeptanz gefunden haben, oder clientseitige Suchwerkzeuge wie die Google-Toolbar dabei eine Rolle. Die Fortschritte auf dem Gebiet der Softwareagenten werden diese Entwicklung unterstützen.

[19] Welcher Klasse man dies konkret zuordnet, dies hängt von der Strenge der jeweiligen Eigenschaftsdefinition ab. Da erst bei einer Implementation die tatsächliche Leistungsfähigkeit getestet werden kann, sei hier auf eine konkrete Zuordnung verzichtet.
[20] z.B. Copernic Agent http://www.copernic.com (16.07.2003)

5. Literatur und Verzeichnisse

[Appleby, 1994] Appleby, S. & Steward, S., "Mobile Software Agents for Control in Telecommunications Networks". BT Technological Journal 12 (2), 104-113, April 1994

[Armstrong et al., 1995] R. Armstrong, D. Freitag, T. Joachims, and T. Mitchell, "Webwatcher: A learning apprentice for the world wide web". In AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments. March 1995

[Brooks, 1991b] Brooks, R. A., "Intelligence without Representation". Artificial Intelligence 47, 139-159. 1991b

[Capurro, 2003] Rafael Capurro, "Einführung in den Informationsbegriff". http://www.capurro.de/infovorl-index.htm (18.06.2003)

[Drogoul, 1995] Alexis Drogoul, "When Ants Play Chess (Or Can Strategies Emerge From Tactical Behaviors?)". From Reaction to Cognition - Fifth European Workshop on Modelling Autonomous Agents in a Multi-Agent World, MAAMAW-93 (LNAI Volume 957)

[Finin, 1991] Finin, T. & Wiederhold, G., "An Overview of KQML: A Knowledge Query and Manipulation Language". Department of Computer Science, Stanford University. 1991.

[Finin, Labrou, 1995] Y. Labrou and T. Finin, "Towards a standard for an Agent Communication Language". Proceeding of the first international conference on Multi-Agent Systems. 1995

[Foner, 1993] Foner, L., "What is an Agent, Anyway? A Sociological Case Study". Agents Memo 93-01, MIT Media Lab, Cambridge, MA. 1993

[Friburger 2002] N. Friburger, D. Maurel, "Textual similarity based on proper names". Mathematical Formal Information Retrieval (MFIR'2002), pp. 155-167. 2002

[Gürer, Lakshminarayan, Sastry, 1998] Denise Gürer, Vinay Lakshminarayan, Ambatipudi Sastry, "An intelligent-agent-based architecture for the management of heterogeneous networks". Information, Telecommunications, and Automation Division, SRI International. 1998

[Huhns, Singh, 1994] Huhns, M. N. & Singh, M. P., "Distributed Artificial Intelligence for Information Systems". CKBS-94 Tutorial, June 15, University of Keele, UK. 1994

[Jones et al., 1998] Steve Jones, Sally Jo Cunningham, Rodger McNab, "An Analysis of Usage of a Digital Library". European Conference on Digital Libraries. 1998

[Kaebling, 1991] Kaebling, L.P., "A situated automata approach to the design of embedded agents". SIGART bulletin, 2(4):85-88

[King, 1995] King, J. A., "Intelligent Agents: Bringing Good Things to Life". AI Expert, February, 17-19. 1995

[Kluge und Menzel, 2002] Stefan Kluge, Gerald Menzel, "Personalisierungstechniken im WWW - Untersuchung der Anforderungen und Möglichkeiten anhand einer prototypischen Umsetzung". HTWK Leipzig. März 2002

[Knoblock 1998] Craig A. Knoblock and Steven Minton, "The ariadne approach to web-based information integration". IEEE Intelligent Systems , 13(5), September/October 1998

[Laurel, 1997] Brenda Laurel, "Interface Agents: Metaphors with Character". Software Agents, pages 67-77. AAAI Press / The MIT Press, Menlo Park, California. 1997

[Lieberman, 1995] Lieberman, H., "Letizia: An Agent that Assists Web Browsing". In Proceedings of IJCAI 95, AAAI Press. 1995

[Liebermann et al., 2001] Henry Lieberman, Elizabeth Rosenzweig And Push Singh, "Aria: An Agent For Annotating And Retrieving Images". IEEE Computer, July 2001, pp. 57-61

[Maes, 1991] Pettie Maes, "The agent network architecture (ANA)". SIAGRT bulletin, 2(4):115-120

[Nwana 1996] Nwana, Hyacinth S., "Software Agents: An Overview". Knowledge Engineering Review, Vol. 11, No 3, pp.1-40. Sept 1996

[Nwana, 1996] Hyacinth S. Nwana, "Software Agents: An Overview". Knowledge Engineering Review, Vol. 11, No 3, pp.1-40. 1996

[Shoham, 1993] Shoham, Y., "Agent-oriented programming". Artificial Intelligence, 60(1):51-92. 1993

[Tripathi, 2002] Anand Tripathi et al., "Active Monitoring of Network Systems using Mobile Agents". In proceedings of Networks 2002, a joint conference of ICWLHN 2002 and ICN 2002, 269-280, 2002

[Wooldridge, Jennings, 1995a] Wooldridge, M. & Jennings, N., "Intelligent Agents: Theory and Practice". The Knowledge Engineering Review 10 (2), 115-152. 1995a

[Zuse, 1999a] Zuse, H., "Geschichte der Programmiersprachen". 1999a. http://www.cs.tu-berlin.de/~zuse (22.06.2003)

Abbildungen

Abbildung 1: drei Informationsebenen und deren Zusammenhang 2-8
Abbildung 2: Leistungsfähigkeit kollaborativer Softwareagenten 2-15
Abbildung 3: Agenten nach autonomen und sozialen Fähigkeiten 2-22
Abbildung 4: agentenbasierte Architektur eines Netzwerkmanagementsystems [Gürer, Lakshminarayan, Sastry, 1998] 2-24
Abbildung 5: Wahrscheinlichkeit der Linkverfolgung durch einen Nutzer 2-38
Abbildung 6: Anteil der Wartung an den Entwicklungskosten [Zuse, 1999a] 2-42
Abbildung 7: gemeinsame und unterschiedliche Interessen zweier Nutzer für die Realisierung eines kollaborativen Filterns 2-53
Abbildung 8: prototypisches Interface des konzipierten Suchwerkzeuges 3-55
Abbildung 9: Schematischer Programmablauf in drei Phasen 3-64
Abbildung 10: Wahrscheinlichkeit für gemeinsames Auftreten zweier Wörter in einem Satz 3-70
Abbildung 11: Termfrequenz-Vektor für den Text aus Beispiel 5 (Auszug) ... 3-72
Abbildung 12: normierter Termfrequenz-Vektor für den Text aus Beispiel 5. 3-72
Abbildung 13: Vektor für Termwahrscheinlichkeit in allgemeinen Texten bezüglich Text aus Beispiel 5 3-73
Abbildung 14: Vektor für den Text aus Beispiel 5: relative Häufigkeit der Terme bezüglich allgemeiner Texte 3-73
Abbildung 15: Phase 1 – Bestimmung des Informationswunsches; Unterscheidung zwischen direkter Eingabe der Suchanfrage und indirektem Informationsbedürfnis 3-76
Abbildung 16: Phase 2 – Personalisierung und Informationsgewinnung; Übergang von der Suchanfrage zu den Ergebnisdokumenten 3-77
Abbildung 17: Binäre Abbildung eines Terms von Suchanfragen; geeignet für das Term-Vektor-Modell 3-79
Abbildung 18: Phase 3 – Informationsausgabe und Lernmethoden; Auswertung und Ranking der Suchergebnisse; Lernmethoden anhand des Anwenderfeedbacks 3-85
Abbildung 19: inhaltlich relevanter Bereich einer Nachrichtenseite für die Dokumentanalyse; Menüs und sonstige herauszufilternde Informationen sind ausgegraut 3-86

Tabellen

Tabelle 1: Suchfunktionen auf syntaktischer Ebene der 3 meistgenutzten Suchmaschinen 3-58

Beispiele

Beispiel 1: Grundstruktur einer KQML-Nachricht 2-19

Beispiel 2: Beispiel einer KQML-Nachricht unter Verwendung von KIF als Wissensrepräsentationssprache 2-19

Beispiel 3: Regel in einer regelbasierten Personalisierung (natürlichsprachlich) 2-51

Beispiel 4: Suchanfrage in Stichwortform 3-57

Beispiel 5: Beispieltext zur Erläuterung der Termfrequenz-Analyse 3-71

Formeln

Formel 1: Cosinus-Ähnlichkeitsmaß für zwei Vektoren $\vec{x}_1, \vec{x}_2$ 3-90

Formel 2: Cosinus-Ähnlichkeitsmaß für zwei normalisierte Vektoren $\vec{x}_1, \vec{x}_2$ ... 3-90