Otto-von-Guericke-Universität Magdeburg



Fakultät für Informatik
Institut für Verteilte Systeme

Studienarbeit

Semantikbasierte Websuche am Beispiel einer Jobsuchmaschine

Verfasser:
Bernd Zuther

18. August 2009

Betreuer:
Prof. Dr.-Ing. habil. Reiner Dumke
Universität Magdeburg
Fakultät für Informatik
Postfach 4120, D–39016 Magdeburg
Germany

Zuther, Bernd:
Semantikbasierte Websuche am Beispiel einer Jobsuchmaschine
Studienarbeit, Otto-von-Guericke-Universität
Magdeburg, 2009.

Abstract

Diese Studienarbeit beschäftigt sich mit der semantischen Websuche. Sie beschreibt die generelle Funktionsweise von Web-Suchmaschinen. Außerdem zeigt sie, dass eine semantische Suche die Genauigkeit der gefunden Suchergebnisse verbessert. Als Beispiel-Implementierung wird dazu die semantische Suchmaschine jobsucher.ch vorgestellt, die während der Praktikumszeit weiterentwickelt wurde.

Inhaltsverzeichnis

Inhaltsverzeichnis **ii**

Abbildungsverzeichnis **iv**

Tabellenverzeichnis **v**

Verzeichnis der Abkürzungen **vi**

1 Einleitung **1**

1.1 Einführung 1

1.2 Ziele der Arbeit 1

1.3 Aufbau der Arbeit 2

2 Stand der Technik **3**

2.1 Information Retrieval 3

2.1.1 Zweck des Information Retrieval 3

2.1.2 Konkreter und problemorientierter Informationsbedarf 3

2.1.3 Arbeitsweise eines Information Retrieval Systems 5

2.1.4 Recall und Precision 6

2.2 Web Information Retrieval 7

2.2.1 Struktur des World Wide Web 8

2.2.2 Unterschiede zwischen Web-Information Retrieval und klassischen Information Retrieval 8

2.2.3 Architektur einer Web-Suchmaschine 12

2.3 Semantische Websuche 14

2.3.1 START 15

2.3.2 Semantik 15

2.3.3 Ontologien 16
2.3.4 Resource Description Framework 17

3 Jobsucher.ch **19**
3.1 Einführung 19
3.1.1 Einordnung des Jobsucher im Gesamtsystem 19
3.1.2 Notwendigkeit einer Ontologie 20
3.2 Projekt-Aufbau 21
3.2.1 Aufgaben des Jobsucher 21
3.2.2 Entwicklungsprozess 22
3.2.3 Maven 26
3.3 Arbeitsweise des Jobsuchers 27
3.3.1 Aufbau des User-Interface 27
3.3.2 Aufbau der Jobsucher-Ontologie 27
3.3.3 Funktionsweise des Jobsuchers 29

4 Auswertung der Ergebnisse **34**
4.1 Auswertung der Suchergebnisse 34
4.1.1 Bestimmung der Precision 34
4.1.2 Bestimmung der Recall 35
4.1.3 Vergleich der Precision mit anderen Suchmaschinen 36
4.2 Bewertung der Ontologie 37

5 Zusammenfassung und Ausblick **38**
5.1 Zusammenfassung 38
5.2 Ausblick 39

Literaturverzeichnis **40**

Abbildungsverzeichnis

2.1 Information Retrieval - Infomation Indexing [Sto07] 6
2.2 Bow-Tie-Struktur des Web ([Lew05]) 8
2.3 Allgemeine Architektur einer Web-Suchmaschine ([RD03]) 12
2.4 Architektur der Suchmaschine AltaVista ([Lew05]) 13
2.5 Bestandteile der Semiotik [Mün09] 15
2.6 Semiotisches Dreieck [Wik09] 16
2.7 Ein einfacher RDF-Graph zur Beschreibung einer Beziehung einer Arbeit über die semantische Websuche und dem Autor Bernd Zuther 17

3.1 Use-Case-Diagramm des Jobagenten 20
3.2 Use-Case-Diagramm des Jobsucher 22
3.3 Agiler Zyklus über schnellen Systemeinsatz [ia09] 23
3.4 Schema des Feature Driven Development - Prozess 25
3.5 Suchfunktionen des Jobsucher 27
3.6 Ausschnitt des Datenbank Schema der Ontologie 28
3.7 High-Level-Architektur für Time Expression 31
3.8 Verteilungsdiagramm Jobsucher und Crawler 32

4.1 Ergebnisse der Stichprobe von 3392 Anzeigen 35
4.2 Vergleich mit anderen Jobsuchmachinen 37

Tabellenverzeichnis

2.1 Arten des Informationsbedarf [Sto07] 4
2.2 Unterschiede zwischen Web-Information Retrieval und klassischen Information Retrieval [Lew05] 12

3.1 Vor- und Nachteile von RDF gegenüber einer relationalen Datenbank . . 28
3.2 Vor- und Nachteile des MVC-Konzept [Geh08] 31

Verzeichnis der Abkürzungen

CIN	Concrete Information Need
DBE	Deskriptorspeicher
DE	Dokumenteneinheiten
IR	Information Retrieval
IRS	Information Retrieval System
POIN	Problem Oriented Information Need
POM	Project Object Model
RDF	Resource Description Framework
SCC	Strongly Connected Core

Kapitel 1

Einleitung

1.1 Einführung

Am 17.06.2009 wird Bernd Z. in den Konferenzraum seiner Firma bestellt. Als nun seine beiden Vorgesetzten, mit einem nichts sagenden Gesichtsausdruck, vor ihm sitzen, schwant ihm nichts Gutes. Vorher ist es noch nie passiert, dass beide gleichzeitig die Zeit gefunden haben mit ihm zu sprechen.

Herr Z. bekam die Nachricht, dass er betriebsbedingt gekündigt wird. Er ist nun in Zeiten der Wirtschaftskrise dazu gezwungen, sich der Herausforderung der Stellensuche zustellen. Sein soziales Netzwerk kann ihm in diesem Moment auch nicht weiterhelfen. Damit ist er gezwungen, sich auf dem klassischen Weg eine neue Stelle zu suchen.

Über die Stellenanzeigen in Zeitungen hat er die Möglichkeit seine Suche beginnen. Doch findet Herr Z. so nur einen Bruchteil an Vakanzen. Um viele Anzeigen zu sichten, muss er sich sehr viele Tageszeitungen besorgen, was für ihn einen enormen finanziellen Aufwand bedeutet.

Kostengünstiger ist es, mit Hilfe von “Yellow Pages” nach Firmen suchen und sich durch deren Webseite über freie Stellen zu informieren. Das bedeutet für Herr Z. nur einen enormen zeitlichen Aufwand. Doch diesen Prozess automatisieren zum Glück für ihn Jobsuchmaschinen und er kann die dadurch gewonnene Zeit effektiver nutzen, um seine Bewerbungsschreiben zu verbessern oder sich genauer über sein neues Einsatzgebiet zu informieren.

Diese Arbeit beschäftigt sich mit der Jobsuchmaschine jobsucher.ch. Diese kann Herr Z. benutzen, um sich über freie Stellen in der Schweiz zu informieren.

1.2 Ziele der Arbeit

Das Ziel dieser Arbeit ist es die Funktionsweise einer Jobsuchmaschine zu erklären und die Analogien zu einem klassischen Information Retrieval System (IRS) aufzuzeigen. Diesbezüglich werden allgemeine Konzepte des Information Retrieval (IR) vorgestellt

und deren Anwendung bei der entwickelten Jobsuchmaschine dargelegt.

Zusätzlich verfolgt diese Arbeit das Ziel zu zeigen, dass durch Verwendung einer Ontologie die Qualität der gefunden Ergebnisse verbessert wird. Denn durch die semantische Jobsuche steigt die Anzahl der für den Stellensuchenden signifikanten Vakanzen.

1.3 Aufbau der Arbeit

Mit allgemeinen Betrachtungen zum Thema IR beginnt Kapitel 2. Es erörtert die Fragen: Welche Probleme Menschen veranlassen nach Informationen zu suchen und welche Prozesse nötig sind, um diese zufinden. Im Anschluss daran werden Metriken vorgestellt, um die Qualität eines IRS zu bewerten. Dann folgt eine Erläuterung der Besonderheiten bei der Websuche und die Architektur einer Websuchmaschine wird vorgestellt. Abschließend wird die semantische Suche als Alternative zur stichwortbasierten Suche erläutert und die Voraussetzungen dafür erklärt.

Das Kapitel 3 dieser Arbeit beschäftigt sich mit der semantischen Jobsuchmaschine jobsucher.ch. Zuerst wird das System beschrieben in dem sich die Suchmaschine befindet. Dann wird an einem Beispiel erklärt, warum es sinnvoll ist bei einer Jobsuchmaschine eine Ontologie zu benutzen. Gefolgt davon wird der Projektaufbau beschrieben. Hier werden die Anforderungen an die Suchmaschine geklärt und der Entwicklungsprozess vorgestellt, in dem sie entstanden ist. Daran schließt ein technischer Einblick in die Funktionsweise der Jobsuchmaschine.

Im Kapitel 4 erfolgt dann eine Auswertung der Jobsuchmaschine hinsichtlich der in Kapitel 2 vorgestellten Metriken. Die Ergebnisse werden bezüglich der Genauigkeit mit anderen Jobsuchmaschinen verglichen. Außerdem folgen Anmerkungen zur Ontologie.

Eine Zusammenfassung der Erkenntnisse dieser Arbeit wird in Kapitel 5 durchgeführt und es erfolgt ein Ausblick auf mögliche Verbesserungen des Jobsuchers.

Kapitel 2

Stand der Technik

2.1 Information Retrieval

Die Anzahl der Wissenschaftler ist in den letzten 150 Jahren deutlich gestiegen. Die daraus resultierende Publikationsaktivität hat zu einer Flut an wissenschaftlichen Publikationen geführt. Durch diese Informationsflut ist es heute für einen Wissenschaftler kaum noch möglich alle Literatur eines Fachgebiets zu überblicken. Aus diesen Grund ist es notwendig, dass man bei der Bearbeitung eines Problem nur auf erforderliche Informationen zugreift, vgl. [Gau05].

2.1.1 Zweck des Information Retrieval

Das Ziel des Information Retrieval (IR) liegt im gezielten Wiederfinden und Nutzbarmachen von Informationen. Das Sammeln und Ordnen ist dabei nur ein Teilgebiet. Wörtlich aus dem Englischen übersetzt heißt Information Retrieval nämlich "Zurückgewinnen von Informationen". Im folgenden wird IR wie folgt definiert:

> "Information retrieval is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers)."[CDM08]

Bevor man Informationen suchen kann, muss also ein Bedürfnis nach Informationen existieren. Dieses Bedürfnis leitet den Benutzer dazu ein Information Retrieval System (IRS) benutzen.

> "An information need is the perceived need for information that leads to someone using an information retrieval system in the first place." [BSC97]

2.1.2 Konkreter und problemorientierter Informationsbedarf

Es existieren zwei mögliche Ausprägungen von Informationsbedürfnissen. Um diese zu verdeutlichen, werden die folgenden Fragestellungen benutzt:

1. Fragetyp A
   (a) Wie heißt die Hauptstadt der Bundesrepublik Deutschland?
   (b) Wer ist der aktuelle Bundespräsident der Bundesrepublik Deutschland?
   (c) Wie lautet die URL der “Otto-von-Guericke” Universität in Magdeburg?
2. Fragetyp B
   (a) Wie hängen Recall und Precision zusammen?
   (b) Wie ist die momentane Wirtschaftskrise zu bewerten?
   (c) Welche Interpretationsmöglichkeiten hat die Figur Peter Pan?

Der Fragetyp A zielt dabei nur auf Übermittlung einer Fakteninformation ab. Der zu Grunde liegende Informationsbedarf ist konkret. Deshalb wird dieser Typ als “Concrete Information Need (CIN)” bezeichnet. Als Sonderfall von CIN lassen sich nach Wolfgang Stock [Sto07] Navigationsprobleme im WWW (dritte Frage) ansehen.

Mit der Angabe eines Faktums lässt sich Fragetyp B hingegen nicht befriedigen. Zur Befriedigung des Informationsbedarfs benötigt man eine größere Sammlung von Texten. Deshalb wird bei ihm vom “Problem Oriented Information Need (POIN)” gesprochen.

Concrete Information Need (CIN)	**Problem Oriented Information Need (POIN)**
Thematische Grenzen sind klar abgesteckt.	Thematische Grenzen sind nicht exakt bestimmbar.
Die Suchfrageformulierung ist durch exakte Terme ausdrückbar.	Die Suchfrageformulierung lässt mehrere terminologische Varianten zu.
Eine Fakteninformation reicht in der Regel aus, um den Bedarf zu decken	In der Regel müssen diverse Dokumente beschafft werden. Ob der Informationsbedarf damit abschließend gedeckt ist,bleibt offen.
Mit der Übermittlung der Fakteninformationen ist das Informationsproblem erledigt.	Mit der Übermittlung der Literaturinformationen wird ggf. das Informationsproblem modifiziert oder ein neuer Bedarf entdeckt.

Tabelle 2.1: Arten des Informationsbedarf [Sto07]

Mit Hilfe von Table 2.1 lassen sich CIN und POIN vergleichen. Beim CIN sind die thematischen Grenzen klar abgesteckt. Hier lassen sich exakte Suchterme formulieren. Eine Information reicht aus um den Informationsbedarf zudecken und nach Übermittlung der

Fakteninformation ist das Problem gelöst.

Anderes ist das beim POIN. Hier sind die thematischen Grenzen nicht klar bestimmt. Deshalb kann man keine exakten Suchterme formulieren und es müssen mehre Dokumente beschafft werden, die das Informationsproblem dann modifizieren oder einen neuen Informationsbedarf wecken. Damit bleibt offen, ob der Informationsbedarf überhaupt gedeckt werden kann.

2.1.3 Arbeitsweise eines Information Retrieval Systems

Nun wird der Begriff Information Retrieval System (IRS) wie folgt definiert:

> "Allgemein wird unter einem Information Retrieval System (IRS) die Gesamtheit aus methodischen Grundlagen und Verfahren verstanden, die Information Retrieval ermöglichen." [Poe06]

Aus dieser Definition heraus stellt sich nun die Frage, was die methodischen Grundlagen und Verfahren eines IRS sind. Nach Wilhelm Gaus [Gau05] werden die folgenden vier Verfahren unterschieden die IR ermöglichen:

1. **Beschaffen und Erfassen**

   Zuerst ist es wichtig alle Dokumente zu beschaffen, die für den Benutzer des IRS wichtig sind. Dabei ist es von Bedeutung keine Dokumente zu übersehen. Denn man spricht sonst von einer unvollständigen Dokumentation[1].

2. **Indexieren**

   Das Erschließen der Dokumente nennt man indexieren, darunter versteht man das Feststellen und Kennzeichnen des Inhalts eines Dokuments. Dies geschieht mit Hilfe sogenannter Deskriptoren. Das können im einfachsten Fall Schlagwörter sein. Beim Indexieren handelt es sich also, um das inhaltliche Erschließen von Dokumenten.

3. **Speichern**

   Beim Speichern werden der Dokumentenspeicher und der Deskriptorenspeicher unterschieden. Im Dokumentenspeicher befinden sich alle Dokumente in ihrem vollen Umfang. Zu welchem Dokument welcher Deskriptor indexiert wurde, speichert der Deskriptorenspeicher.

[1] Im Englischen wird Dokumentation als Information Retrieval bezeichnet, vgl. [Gau05]

4. **Recherchieren**

   Die Recherche ist das gezielte Suchen und Wiederauffinden von Dokumenten. Der interessierende Sachverhalt wird als allgemeine und verbale Suchfrage bezeichnet. Wenn diese durch Deskriptoren ausgedrückt wird, bildet sie die formale Suchfrage. Mit dieser Suchfrage wird dann im Deskriptorspeicher gezielt nach Dokumenten gesucht.

Die meisten IRS arbeiten mit dem Exact Match[2]-Verfahren, vgl. [Sto07]. Dazu muss die Suchsyntax beherrscht und Suchargumente fehlerfrei eingegeben werden. Damit gibt der Nutzer durch die Anfrageformulierung vor, wie der Retrievalprozess abläuft.

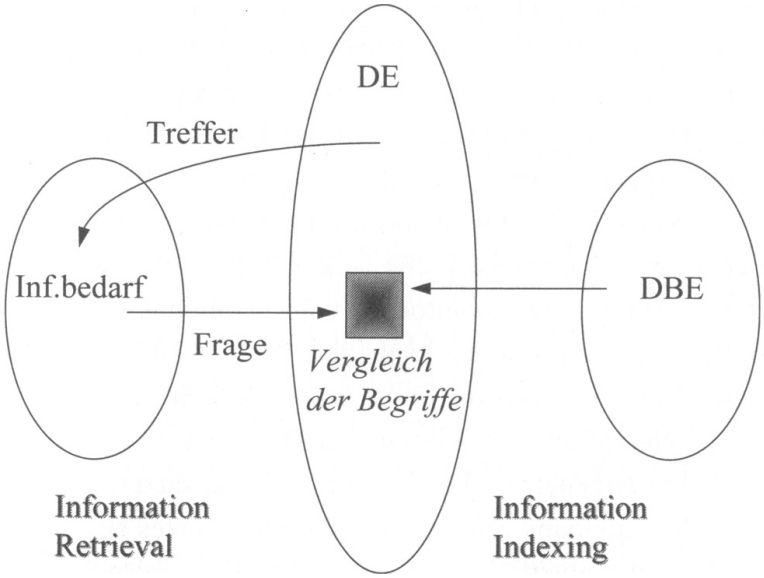


Abbildung 2.1: Information Retrieval - Infomation Indexing [Sto07]

Das Zusammenspiel zwischen Dokumentenspeicher und Index wird in Abbildung 2.1 dargestellt. Der Nutzer stellt eine Frage an die Dokumenteneinheiten (DE). Dort werden dann die Begriffe oder Worte verglichen. Wortbasierte Systeme erfassen dabei ausschließlich Zeichenfolgen, die als Suchargumente verwendet und mit dem Deskriptorspeicher (DBE) verglichen werden. Aus der Menge aller DE werden vom IRS alle Dokumente als Treffer herausgefiltert, die der Suchanfrage entsprechen und dem Nutzer als Ergebnisse angezeigt.

2.1.4 Recall und Precision

Die Größe aller DE lässt sich ohne Probleme ermitteln, in dem man die Anzahl aller Dokumente bestimmt. Doch sagt dies noch nichts über die Güte eines IRS aus. Wie in Abschnitt 2.1.1 beschrieben wurde, ist der Zweck IR aber nicht das Speichern von Dokumenten sondern das Wiederauffinden von Dokumenten.

[2] Exact Match liefert nur die Treffer, die der Suchanfrage konkret entsprechen.

Wolfgang G. Stock [Sto07] geht sogar noch weiter und sagt, dass das Ziel des IR das Finden von möglichst relevanten bzw. pertinenten Dokumente zu Befriedung eines Informationsbedarfs ist. Denn Dokumente, die nichts mit dem Thema zu tun haben, sind störend. Kennzahlen für die Qualität eines IRS sind die beiden Metriken "Recall" und "Precision". Es wird bei der Berechnung von drei Mengen ausgegangen:

- $a =$: gefundene relevante Dokumente,
- $b =$: nichtrelevante Dokumenten, die in der Treffermenge enthalten sind (Ballast),
- $c =$: relevante Dokumente, die nicht gefunden wurden (Verlust).

Die *Recall* (Vollständigkeit) berechnet sich aus dem Quotient aus der Anzahl der gefundenen relevanten Dokumenten und der Gesamtzahl der relevanten Dokumente.

$$Recall = \frac{a}{(a+c)} \tag{2.1}$$

$$Recall = \frac{\text{gefundene relevante Dokumente}}{\text{Gesamtanzahl der relevanten Dokumente}} \tag{2.2}$$

Die *Precision* (Genauigkeit) errechnet sich aus dem Quotient aus der Anzahl der gefundenen relevanten Dokumente und der Anzahl aller gefundenen Dokumente.

$$Precision = \frac{a}{(a+b)} \tag{2.3}$$

$$Precision = \frac{\text{gefundene relevante Dokumente}}{\text{Anzahl aller gefundenen Dokumente}} \tag{2.4}$$

Die Werte a und b lassen sich experimentell bestimmen, damit ist die Precision genau messbar. Ein reines Konstrukt stellt hingegen der Recall da. Denn c ist nicht direkt messbar.

2.2 Web Information Retrieval

Das Web ist aus unseren heutigen Alltagsleben kaum noch wegzudenken. Ausserdem beinhaltet es eine dem menschlichen Vorstellungsvermögen längst übersteigende und ständig anwachsende Menge an Informationen, vgl. [PH08].

Dirk Lewandowski [Lew05] stellt die These auf, dass es ökonomisch nicht lohnt eine hohe Vollständigkeit der Indexierung der Web-Inhalten zu erreichen. Denn es werden nur wenige Dokumente häufig nachgefragt und andere nur sehr selten.

2.2.1 Struktur des World Wide Web

Die Struktur des Web wird in Abbildung 2.2 gezeigt und ist das Ergebnis der Verlinkung von Dokumenten untereinander. Sie wird auch als Fliegen-Struktur bezeichnet und besteht im Inneren aus Dokumenten, die stark untereinander verbunden sind, dem Strongly Connected Core (SCC). Dann gibt es einen Bereich IN mit Dokumenten die auf den SCC verweisen und einen Bereich OUT auf dem die Dokumente des SCC verweisen.

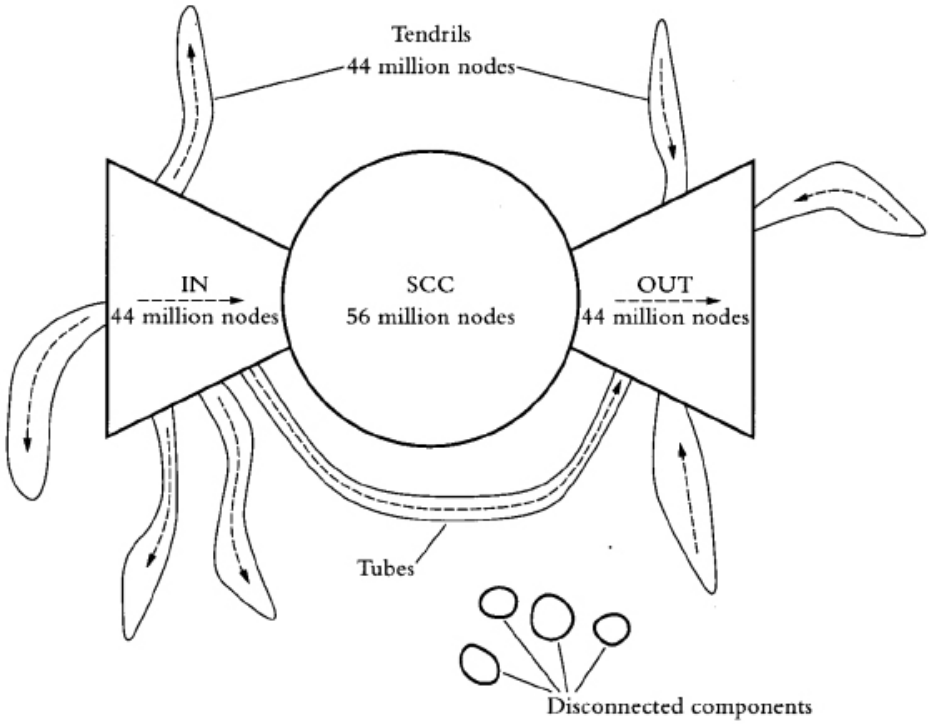


Abbildung 2.2: Bow-Tie-Struktur des Web ([Lew05])

Verbindungen zwischen dem IN- und Out-Bereich existieren nur vereinzelt und werden als “tubes” bezeichnet. Neben den Bereichen die direkt miteinander verbunden sind, existiert noch der Bereich, der sog. “tendrils”. Bei dem die Dokument zwar mit den großen drei Subgraphen verbunden, aber trotzdem irrelevant isoliert sind. Zuletzt gibt es noch den letzten Bereich der “disconnected components”. Dieser besteht aus Dokumenten, die mit keinem Subgraphen verbunden sind.

Nach Lewandowski [Lew05] ergeben sich aus der Fliegen-Struktur zwei Schlüsse. Dokumente die sich im IN-Bereich des Web befinden, lassen sich deutlich schwieriger aufspüren als Dokumente im Kernbereich oder im Out-Bereich. Ausserdem zeigt der Bereich der “disconnected components” deutlich, dass nicht alle Dokumente erfasst werden können.

2.2.2 Unterschiede zwischen Web-Information Retrieval und klassischen Information Retrieval

Die Unterschiede zwischen Web-Information Retrieval und dem klassischen Information Retrieval lassen sich in vier Klassen unterteilen. Eine ausführliche Auseinandersetzung

mit diesen Klassen findet Tabelle 2.2 und man entnimmt ihr, dass sich die Unterschiede auf

- dem Korpus der Dokumente,
- hinsichtlich der Inhalte,
- dem Nutzer und
- der Eigenarten des IRS selbst

beziehen. Nachfolgend werden nur die Merkmale dieser Klassen in Bezug auf das Web-Information Retrieval besprochen.

Im Web liegen die Dokumente in mehren Sprachen vor. Außerdem können die Dokumente in unterschiedlichen Formaten vorkommen und werden oft aufgeteilt, wo durch die Länge variiert. Wie im vorangegangen Abschnitt schon besprochen sind die Dokumente miteinander verbunden. Weiterhin hat man mit Spam als Problem zu kämpfen.

Wenn man sich nun den Inhalten zuwendet, kann man folgende Aussagen treffen. Die genaue Datenmenge ist nicht exakt bestimmbar. Aus diesem Grund kann man das Web nicht vollständig indexieren. Darüber hinaus können Dokumente mehrfach vorkommen und teils sogar in unterschiedlichen Versionen.

Die Anfragen an das IRS sind aufgrund der heterogenen Informationsbedürfnisse sehr unterschiedlich. Zudem haben die Benutzer oft nur geringe Kenntnis über die angebotenen Suchfunktionen.

Aus diesen Gründen muss das IRS auch möglichst einfach und intuitiv zu bedienen sein, wodurch die Suchfunktionen meistens beschränkt sind. Bei großen Treffermengen ist ein Relevance Ranking nötigt, um die Ausgabe zu sortieren. Die indexierten Dokumente haben nur eine schwache Strukturierung. Deshalb eignen sie sich nur bedingt für eine Feldsuche[3].

[3] Feldsuche bedeutet, daß vorgegeben wird, in welchem Teil eines Dokumentes gesucht werden soll.

Unterscheidungsmerkmale	**Web**	**Klassische Datenbanken**
Merkmale des Dokumentenkorpus		
Sprachen	Dokumente liegen in einer Vielzahl von Sprachen vor; aufgrund der Volltexterschließung keine einheitliche Erschließung über Sprachgrenzen hinweg.	Einzelne Sprache oder Dokumente liegen in vorherdefinierten Sprachen vor; Erschließung von Dokumenten verschiedener Sprachen mittels einer einheitlichen Indexierungssprache.
Medienarten	Dokumente in unterschiedlichen Formaten.	Dokumente liegen in der Regel in nur einem Format vor.
Länge und Granularität der Dokumente	Länge der Dokumente variiert, große Dokumente werden oft aufgeteilt.	Länge der Dokumente variiert innerhalb eines gewissen Rahmens; pro Dokument eine Dokumentationseinheit.
Spam	Problem der von den Suchmaschinen unerwünschten Inhalte.	Beim Aufbau der Datenbank wird vorab definiert, welche Dokumente erschlossen werden.
Hyperlink-Struktur	Dokumente sind miteinander verbunden.	Dokumente sind in der Regel nicht miteinander verknüpft; keine Notwendigkeit, aus Verlinkungsstrukturen auf die Qualität der Dokumente zu schließen.
Inhalte		
Datenmenge / Größe des Datenbestands	Genaue Datenmenge nicht bestimmbar; keine vollständige Indexierung möglich.	Genaue Datenmenge aufgrund formaler Kriterien bestimmbar.

Abdeckung des Datenbestands	Abdeckung der Zielmenge unklar.	Abdeckung gemäß dem bei der Planung der Datenbank gesteckten Ziel in der Regel vollständig.
Dubletten	Dokumente können mehrfach / vielfach vorhanden sein; teils auch in unterschiedlichen Versionen.	Dublettenkontrolle bei der Erfassung der Dokumente. Versionskontrolle in der Regel nicht notwendig, da jeweils eine endgültige Fassung existiert und diese in die Datenbank eingestellt wird.
Nutzer		
Art der Anfragen	Aufgrund heterogener Informationsbedürfnisse der Nutzer sehr unterschiedlich.	Genaue Zielgruppe mit klarem Informationsbedürfnis.
Ill-formed queries	Geringe Kenntnis der Nutzer über angebotene Suchfunktionen / Recherche allgemein.	Nutzer sind mit der jeweiligen Abfragesprache vertraut.
IR-System		
Interface	Einfache, intuitiv bedienbare Interfaces für Laien-Nutzer.	Oft komplexe Interfaces; Einarbeitung notwendig.
Ranking	Relevance Ranking aufgrund der großen Treffermengen notwenig.	Relevance Ranking aufgrund genau formulierter Suchanfragen und dadurch geringerer Treffermengen meist nicht nötig.
Suchfunktionen	beschränkte Suchfunktionen	komplexe Abfragesprachen

Modifikation der Suche	In der Regel nur Möglichkeiten zur weiteren Einschränkung der Suchanfrage.	Umfangreiche Modifikationsmöglichkeiten
Strukturierung der indexierten Dokumente	schwache Strukturierung; Feldsuche nur bedingt für die Recherche geeignet.	starke Strukturierung; Suche innerhalb einzelner Felder gut für die Recherche geeignet.
Auswahl der Dokumente	Abgesehen vom Ausschluss von Spam keine weitere Auswahlkriterien.	Klare Auswahlkriterien werden schon bei der Planung der Datenbank bestimmt.

Tabelle 2.2: Unterschiede zwischen Web-Information Retrieval und klassischen Information Retrieval [Lew05]

2.2.3 Architektur einer Web-Suchmaschine

Eine sehr vereinfachte Architektur einer Web-Suchmaschine wird in Abbildung 2.3 darstellt. Sie besteht aus einer Benutzeroberfläche, die zur Darstellung von Suchergebnissen, Werbung und der Suchinitialisierung dient. Die Oberfläche kann vom Web aus aufgerufen werden und stellt die Ergebnisse im Web dar.

Suchanfragen werden an die Datenbasis der Suchmaschine übermittelt und diese übermittelt dann Ergebnisse an die Benutzeroberfläche. Die Datenbasis wird dann durch frequentiertes Durchsuchen des Webs verändert bzw. erweitert.

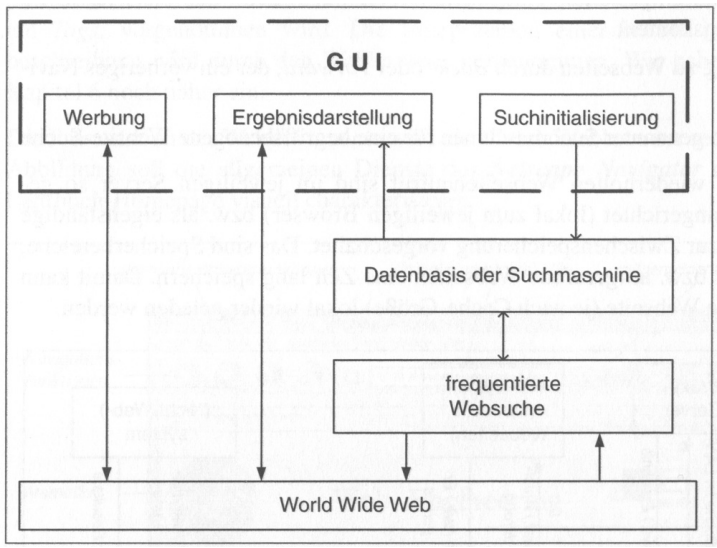


Abbildung 2.3: Allgemeine Architektur einer Web-Suchmaschine ([RD03])

Die Abbildung 2.4 findet man sowohl in [Lew05] als auch in [Sto07]. Sie beschreibt dem

Aufbau einer algorithmischen Suchmaschine[4] prototypisch am Beispiel von AltaVista[5].

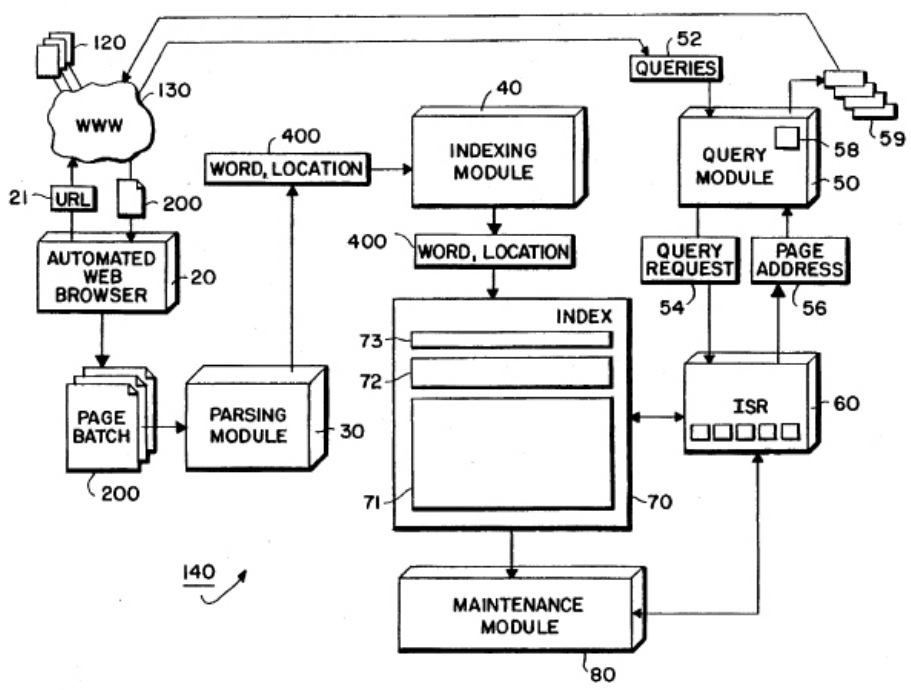


Abbildung 2.4: Architektur der Suchmaschine AltaVista ([Lew05])

Deutlich zu erkennen sind dabei, dass man die im Abschnitt 2.1.3 besprochenen Verfahren "Beschaffen und Erfassen", "Indexieren", "Speichern" und "Recherche", welche sich in den Bestandteilen in den folgenden Bestandteilen wiederfinden:

1. **Automated Web Browser (Crawler)**

   Der Automated Web Browser wird in der Regel als Crawler bezeichnet und hat die Aufgabe Dokumente aufzufinden. Er verfolgt dabei die Hyperlinks innerhalb bereits bekannter Dokumente. Dieser Vorgang findet kontinuierlich statt.

2. **Parsing Module (Syntaxanalyse)**

   Dokumente werden vom Parsing Module in indexierbare Einheiten zerlegt und deren Vorkommen innerhalb des Dokuments verzeichnet.

3. **Indexing Module (Indexierer)**

   Die Wort-Speicherstellen-Paare speichert dann das Indexing Module. Dabei

[4] Algorithmischen Suchmaschinen durchsuchen das Web automatisch und erfassen die gefundenen Dokumente in einer eigenen Datenbank

[5] Lewandowski bemerkt noch, dass die Suchmaschine AltaVista in der dargestellten Form nicht mehr existiert. Diese greift mittlerweile auf die Yahoo-Suchmaschine zurück. Er bemerkt, aber auch das der Aufbau bei allen Suchmaschinen ähnlich ist, aber nicht so genau dokumentiert ist.

werden zwei Indizies erstellt. Es werden den Wörtern die Nummern der Dokumente zugeordnet, in denen diese Vorkommen. Und umgekehrt werden den Dokumentennummer die Wörter zugeordnet. So können einerseits alle Dokumente in denen ein bestimmtes Wort vorkommt ermittelt werden und anderseits alle in einem Dokument vorkommenden Worte.

4. **Query Module (Abfragemodul)**

   Vom Query Module werden die eigegebenen Suchanfragen in eine verarbeitbare Form umgesetzt. Dabei erfolgt eine Auslösung von Befehlen und Operatoren, dass Sie entsprechend im Index abgefragt werden können.

5. **Index Stream Readers (ISR)**

   Die ISR gleichen die umgesetzte Suchanfrage mit dem Index ab und geben die passenden Dokumente an das Query Module zurück.

6. **Index**

   Der Index gewährleistet schnellen Zugriff auf Daten, welche in der Regel üblicherweise sequentiell auf einem Speichermedium verwaltet werden. Im ungünstigsten Fall müsste bei einer Suchanfrage der komplette Datenbestand durchsucht werden. Die Indexstruktur ermöglichst es eine aufwändige Suche zu vermeiden und die Position des Datensatzes schnell zu bestimmen.

7. **Maintenance Module (Datenpflege)**

   Die kontinuierliche Index-Aktualisierung bei laufendem Betrieb und das Aussondern der Dubletten aus dem Index wird vom Maintenance Module übernommen.

2.3 Semantische Websuche

Die beiden Autoren Pascal Hitzler [PH08] und Jens Graupmann [Gra07] beschreiben dass Web-Suchmaschinen ein Problem haben, denn ihre Genauigkeit ist schlecht. Informationen die zwar verfügbar sind, werden in der Fülle der verfügbaren Dokumenten einfach nicht mehr gefunden. Graupman vergleicht dieses Phänomen, sogar mit der “Suche nach der Nadel im Heuhaufen”.

Die meisten Web-Suchmaschinen liefern immer häufiger nur “Mainstream” -Ergebnisse. Betrachtet man zum Beispiel die folgende Suchanfrage: “Britney Spears ring tone”. Auf diese Anfrage sollte jede Suchmaschine eine befriedigende Antwort finden. Die Ergebnisse bei einer Suche der Anfrage “Britney Spears parents” sind hingehen nicht so befriedigend.

Der menschliche Nutzer ist in der Lage die Bedeutung einer Information auf einer Webseite problemlos zu erfassen. Die Information in eine andere Darstellungsform transformieren und mit anderen Informationen in Beziehung setzen. Eine Maschine ist

dazu in der Regel nicht in der Lage, vgl. [PH08] und [Gra07].

Um die Genauigkeit einer Web-Suchmaschine zu verbessern, muss man die stichwortbasierte Suche durch eine inhaltliche Suche ersetzen.

2.3.1 START

START[6] bezeichnet sich selbst, als weltweit erstes webbasiertes Fragen-Antworten-System. Es ging im Dezember 1993 online und wurde von Boris Katz am MIT Computer Science and Artificial Intelligence Laboratory entwickelt. START verdeutlich damit, wie eine inhaltliche Suche aussehen kann.

Das System kann englische Fragen beantworten. Stellt man ihm zum Beispiel die Suchanfrage "who is hans rudolf merz", antwortet das System wie folgt:

"Hans-Rudolf Merz (born 10 November 1942) is a Swiss politician of the Free Democratic Party (FDP/PRD) and member of the Swiss Federal Council (since 2004). He is the head of the Federal Department of Finance (the Swiss finance minister) and President of the Swiss Confederation for 2009."

Will man nur das Geburtsdatum wissen, stellt man folgende Anfrage "when was hans rudolf merz born" und das System antwortet mit:

"Hans-Rudolf Merz was born on November 10, 1942."

START analysiert die Bedeutung einer Suchanfrage und liefert die passende Antwort darauf.

2.3.2 Semantik

Nun ist ein Ausflug in die Semiotik nötig, um den Begriff Bedeutung aus linguistischer Sicht zu klären und wie er sich von anderen Gebieten abgrenzt. Die Semiotik wird in die drei Teilgebiete Syntax, Semantik und Pragmatik aufgeteilt. Wie sich die einzelnen Teilgebiete von einander abgrenzen, zeigt Abbildung 2.5.

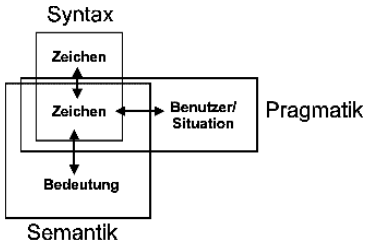


Abbildung 2.5: Bestandteile der Semiotik [Mün09]

[6] http://start.csail.mit.edu/ - 13.06.2009

Im Zentrum aller drei Disziplinen steht das Zeichen. Die Syntax beschäftigt sich mit der Menge aller Regeln zur Strukturierung von Zeichen. Mit der Bedeutung von Wörtern, Phrasen oder Symbolen beschäftigt sich die Semantik. Abschließend ist die Pragmatik die Lehre vom Gebrauch von Sprache in unterschiedlichen Situationen, vgl. [PH08] und [Mün09].

Nach diesem kurzen Exkurs in die Semiotik wird der Begriff Semantik wie folgt definiert:

> "Die Semantik (Bedeutungslehre) ist ein Teilgebiet der Sprachwissenschaft (Linguistik), das sich mit Sinn und der Bedeutung von Sprache beziehungsweise sprachlichen Zeichen befasst." [TP06]

2.3.3 Ontologien

Die grundlegende Aufgabe der Informationsverarbeitung ist es Ausschnitte aus der realen Welt in eine geeignete Darstellungsform so zu transformieren, dass sie mit Hilfe des Computer manipuliert werden kann.

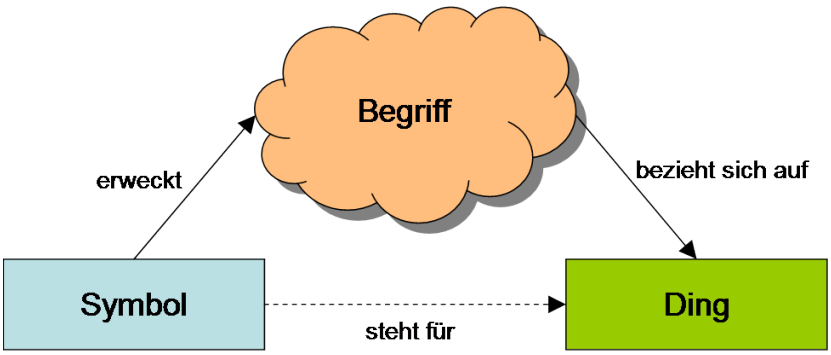


Abbildung 2.6: Semiotisches Dreieck [Wik09]

Odgen und Richardson beschreiben Beziehungen zwischen einer Darstellung, realen Objekten und damit verbundenen Erwartungen. Abbildung 2.5 zeigt nun das sogenannte "Semiotisches Dreieck". Ausgangspunkt ist das Symbol, dass für einen bestimmten Begriff steht und somit bestimmte Erwartungen erweckt. Ein Begriff bezieht sich immer auf ein Objekt in der realen Welt. Dadurch besteht auch eine indirekte Verbindung zwischen einem Symbol und einem Ding. Damit steht ein Symbol auch für ein Ding in der realen Welt, vgl. [Stu09].

Ontologien werden nun benutzt, um diese Beziehungen zu beschreiben. Sie wurden im Umfeld der Künstlichen Intelligenz entwickelt, um Wissen einer Domäne formal zu repräsentieren. Außerdem können sie unabhängig von Programmen wiederverwendet werden und beschreiben die Konzepte und Beziehungen innerhalb einer Wissensdomäne.

Dadurch helfen sie einer Maschine Bedeutungen zu verstehen. Nach Andreas Blumauer und Tassilo Pellegini [TP06] werden sie entwickelt und dazu eingesetzt, um

- den Datenaustausch zwischen Programmen zu ermöglichen,
- verschiedene Wissensrepräsentationsformen zu vereinheitlichen und zu übersetzen,
- Services zur Unterstützung von Wissensarbeitern zu entwickeln,
- Theorien abzubilden,
- die Semantik strukturierter und semi-strukturierter Information auszudrücken,
- die Kommunikation zwischen Menschen zu unterstützen und zu erleichtern.

2.3.4 Resource Description Framework

Das Resource Description Framework (RDF) ist eine formale Sprache für die Beschreibung von Ontologien. Im Gegensatz zu HTML und XML geht es nicht um die konrrekte Darstellung von Dokumenten, sondern um die Kombination und Weiterverarbeitung von Informationen. RDF wird als Darstellungsformat für das Semantic Web angesehen, vgl. [PH08].

Durch das RDF wird ein gerichteter Graphen beschrieben, der aus einer Menge von Knoten besteht. Diese sind durch gerichtete Kanten verbunden. Sowohl die Knoten als auch die Kanten sind mit einem eindeutigen Bezeichner beschriftet. Für die Bezeichnung aller Ressourcen werden grundsätzlich URIs[7] verwendet, auch wenn die Ressourcen im Allgemeinen nicht online abgerufen werden.



Abbildung 2.7: Ein einfacher RDF-Graph zur Beschreibung einer Beziehung einer Arbeit über die semantische Websuche und dem Autor Bernd Zuther

Ein einfaches Beispiel eines Graphen aus zwei Knoten und einer Kante beschreibt Abbildung 2.7. Der Graph wird gebildet durch einen Anfangspunkt (**http://example.org/SemantikbasierteWebsuche**), die Beschriftung der Kante (**http://example.org/Geschrieben**) und einen Endpunkt (**http://zuther.ch/Bernd**). Eine Relation zwischen den Knoten wird durch ein Tripel aus Subjekt, Prädikat und Objekt beschrieben. Durch solche Tripel wird ein RDF-Graph vollständig beschreiben.

Es gibt verschiedene Darstellungsformen für die Serialisierung von RDF. Die gebräuchlichste Schreibweise basiert auf XML. Denn es gibt für nahezu jede Programmiersprache einen XML-Parser. Durch den hierarchisch Aufbau von XML ist auch die Kodierung der RDF-Tripel hierarchisch. Der Graph aus Abbildung 2.7 wird mit Hilfe von XML in Listing 2.1 beschrieben.

[7] URIs sind einer Verallgemeinerung von URLs (Uniform Resource Locator), die zum Zugriff auf ein Online-Dokument verwendet werden

```
1 <?xml version="1.0">
2 <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3          xmlns:ex= "http://example.org/">
4   <rdf:Description rdf:about="http://example.org/SemantikbasierteWebsuche">
5     <ex:VerlegBei>
6       <rdf:Description rdf:about="http://springer.org/SemanticWeb" />
7     </ex:VerlegBei>
8   </rdf:Description>
9 </rdf:RDF>
```

Nach der optionalen Angabe der XML-Version beginnt das Dokument mit einem Knoten vom Typ **rdf:RDF**. Dieses Element wird als Wurzel eines RDF/XML-Dokument verwendet. Innerhalb dieses Elements findet man die Kodierung des Tripels. Subjekt und Objekt werden dabei von durch Elemente des Typs **rdf:Description** beschrieben. Durch das XML-Attribut **rdf:about** wird der Bezeichner des entsprechenden Knotens angeben. Das Prädikat wird direkt als Element **ex:VerlegBei** kodiert.

Kapitel 3

Jobsucher.ch

3.1 Einführung

Wie Herr Z. kommt jeder Mensch früher oder später an einem Punkt in seinen Leben, an dem er sich eine Arbeitsstelle suchen muss, sei es durch eine Kündigung oder den Abschluss seiner Ausbildung. An diesem Punkt wird man sicherlich eine Jobsuchmaschine benutzen, um eine möglichst große Anzahl an Vakanzen zu finden. Das Suchen nach einer freien Stelle spiegelt dabei ein konkretes Informationsbedürfnis wieder.

Bei der Arbeitsplatzsuche ist die eigentliche Suche nach einer Vakanz jedoch nur ein Teilprozess. Hat man eine offene Stelle gefunden, muss man sich noch auf die Stelle bewerben, um sie zu besetzen. Dazu ist es notwendig eine Bewerbung und einen Lebenslauf zu erstellen. Gerade wenn man das erste Mal eine Bewerbung schreibt, wird man in viele Fettnäpfchen treten. Um diese zu vermeiden entwickelt die x28 AG[1] Web-Applikationen, die die Stellensuchenden bei dem gesamten Bewerbungsprozess unterstützen.

3.1.1 Einordnung des Jobsucher im Gesamtsystem

Der Jobsucher[2] ist eine Komponente des Jobagenten[3], der den Prozess der Stellensuche vereinfacht, die Qualität von Bewerbungen verbessert, hilft diese zu organisieren und sich auf Fragen im Bewerbungsgespräch vorzubereiten. Abbildung 3.1 zeigt dazu ein Use-Case-Diagramm dieses Systems. Alle Anwendungsfälle sind in einer eigenen Web-Applikation umgesetzt, die mit einem einzigen Login verbunden werden, realisiert wurde das Single Sign On durch einen Central Authentication Service[4].

Dem Arbeitsuchenden ist gerade am Anfang seiner beruflichen Laufbahn seine Eignung oftmals nicht klar. Deshalb kann er mit Hilfe des Careertools[5] ein Persönlichkeitsprofil anlegen und sich anhand seiner persönlichkeitsspezifischer Merkmale Berufe anzeigen lassen, die zu seinem Profil passen und gezielt nach diesen Stellen suchen.

1 http://www.x28.ch - 20.06.2009
2 http://www.jobsucher.ch - 13.06.2009
3 http://www.jobagent.ch - 13.06.2009
4 http://www.jasig.org/cas - 13.06.2009
5 http://www.careertool.ch - 13.06.2009

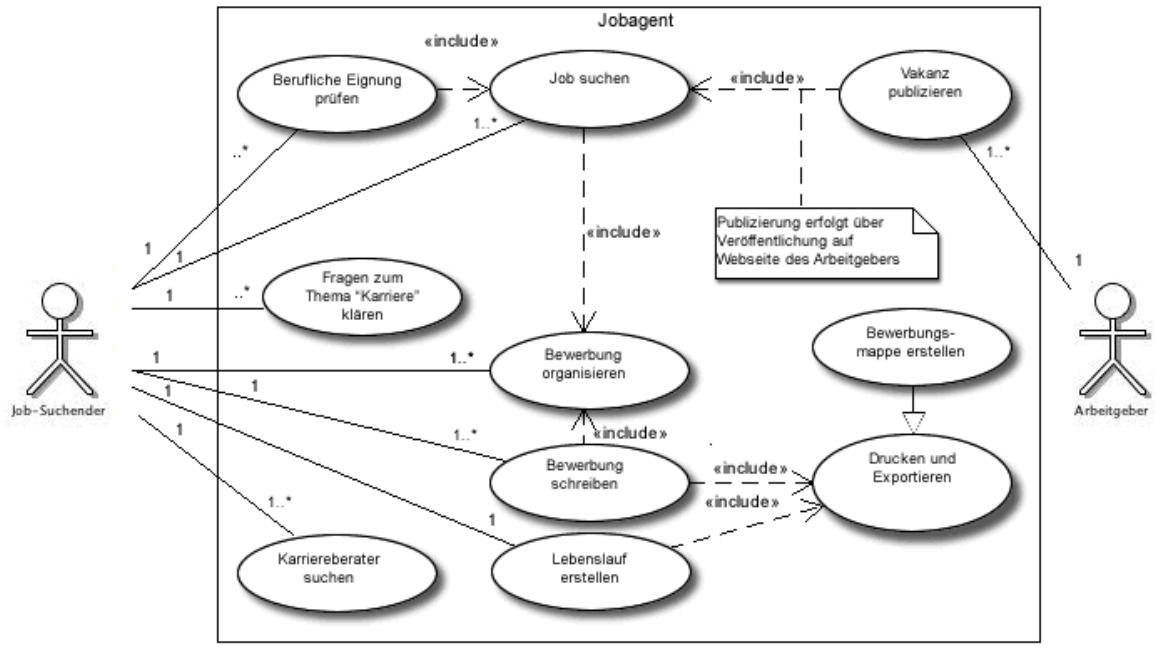


Abbildung 3.1: Use-Case-Diagramm des Jobagenten

Der Jobsucher wird dann für die eigentliche Stellensuche benutzt. Findet der Arbeitssuchende nun eine passende Vakanz, hat er die Möglichkeit sich mit Hilfe des Karriereratgebers[6] und des Karriereglossars[7] über Themen, wie Bewerbungsgespräche, Gehalt oder allgemeine Begriffe zu informieren. Wenn ihm diese Informationen nicht ausreichen, benutzt er das Tool Karriereberater[8], um gezielt nach Personen zu suchen, die ihn unterstützen können.

Beim Erstellen der Bewerbungsmappe unterstützen den Benutzer die Tools Bewerbungsschreiber[9] und Lebenslaufersteller[10]. Mit dem Bewerbungsorganisator[11] [12] hat er dann die Möglichkeit seine Bewerbungen zu katalogisieren und seine Bewerbungsbemühungen zu dokumentieren.

3.1.2 Notwendigkeit einer Ontologie

Im Kapitel 2.1.4 wird über Metriken gesprochen, mit denen Aussagen über die Güte eines IRS getroffen werden. Die wohl wichtigste Metrik eines IRS ist die *Precision* (Genauigkeit), die mit Hilfe von Formel 2.3 berechnet wird. Denn bei Betrachtung der Suchergebnisse wird sie vom Benutzer intuitiv gemessen.

6 http://www.karriereratgeber.ch - 13.06.2009
7 http://www.karriereglossar.ch - 13.06.2009
8 http://www.karriereberater.ch - 13.06.2009
9 http://www.bewerbungsschreiber.ch - 13.06.2009
10 http://www.lebenslaufersteller.ch - 13.06.2009
11 http://www.bewerbungsorganisator.ch - 13.06.2009
12 Dazu existiert eine Publikation von Martin Sturm [Stu08], auf die hiermit verwiesen werden

Die Variable a repräsentiert die gefunden relevanten Dokumente und die Variable b repräsentiert die nicht relevanten Dokumente. Aus der Formel 2.3 wird folgender Zusammenhang abgeleitet, Precision $\sim a$ und Precision $\sim \frac{1}{b}$. Der Einfluss der Proportionalität bewirkt, dass bei einer Erhöhung der Anzahl gefundener relevanten Dokumente sich auch die Precision erhöht. Umgekehrt bewirkt eine Verminderung der gefundenen nicht relevanten Dokumente auch eine Erhöhung der Precision.

Natürlich möchte man auch bei einer Suchanfrage an eine Jobsuchmaschine nur pertinente Stellen als Treffer angezeigt bekommen. Denn Treffer sind ärgerlich, die nichts mit der gewünschten Stelle zu tun haben.

Sucht man in einer klassischen Jobsuchmaschine nach der Stelle "CEO", dann wird diese mit Sicherheit Resultate liefern, in den die drei Buchstaben "CEO" enthalten sind. Damit auch den "CEO-Assistenten". Ein CEO und ein CEO-Assistent haben jedoch ein anderes Anforderungsprofil und ein Geschäftsführer möchte sicher nicht als Assistent arbeiten.

Durch eine semantischen Suche wird der CEO-Assistent über sogenannte Antonyme[13] ausgeschlossen. Ausserdem gibt es noch andere Begriffe, die die gleiche Bedeutung haben, wie z.B. Konzernleiter, Geschäftsführer oder Managing Partner. Diese werden bei einer semantischen Suche über Synonyme[14] mit gesucht.

Die Proportionalitätsbeziehungen der Precision wird bei einer semantischen Suche dazu genutzt, die Precision zu erhöhen, in dem man die Anzahl der gefundenen relevanten Stellen durch die Einführung von Synonymen erhöht. Ausserdem wird durch eine Reduktion der gefundenen nicht relevanten Stellen durch Antonyme auch eine Erhöhung der Precision bewirkt.

3.2 Projekt-Aufbau

3.2.1 Aufgaben des Jobsucher

Die zentrale Aufgabe des Jobsucher ist natürlich die Darstellung der Suchergebnisse einer Suchanfrage auf eine bestimmte Stelle. Selbstverständlich erfüllt er darüber hinaus noch andere Anforderungen, welche in Abbildung 3.2 dargestellt werden. Dazu gehören:

- Die Ergebnisse einer Suchanfrage als RSS-Feed darstellen.
- In regelmässigen Intervallen dem Benutzer via E-Mail über freie Stellen zu informieren.
- Interessante Stellen im Bewerbungsorganisator vormerken.
- Empfehlungen für prämierte Stellen senden und erhalten.

[13] Antonyme sind in der Sprachwissenschaft Wörter mit gegensätzlicher Bedeutung.
[14] Zwei Wörter sind synonym, wenn sie die gleiche (ähnliche) Bedeutung haben.

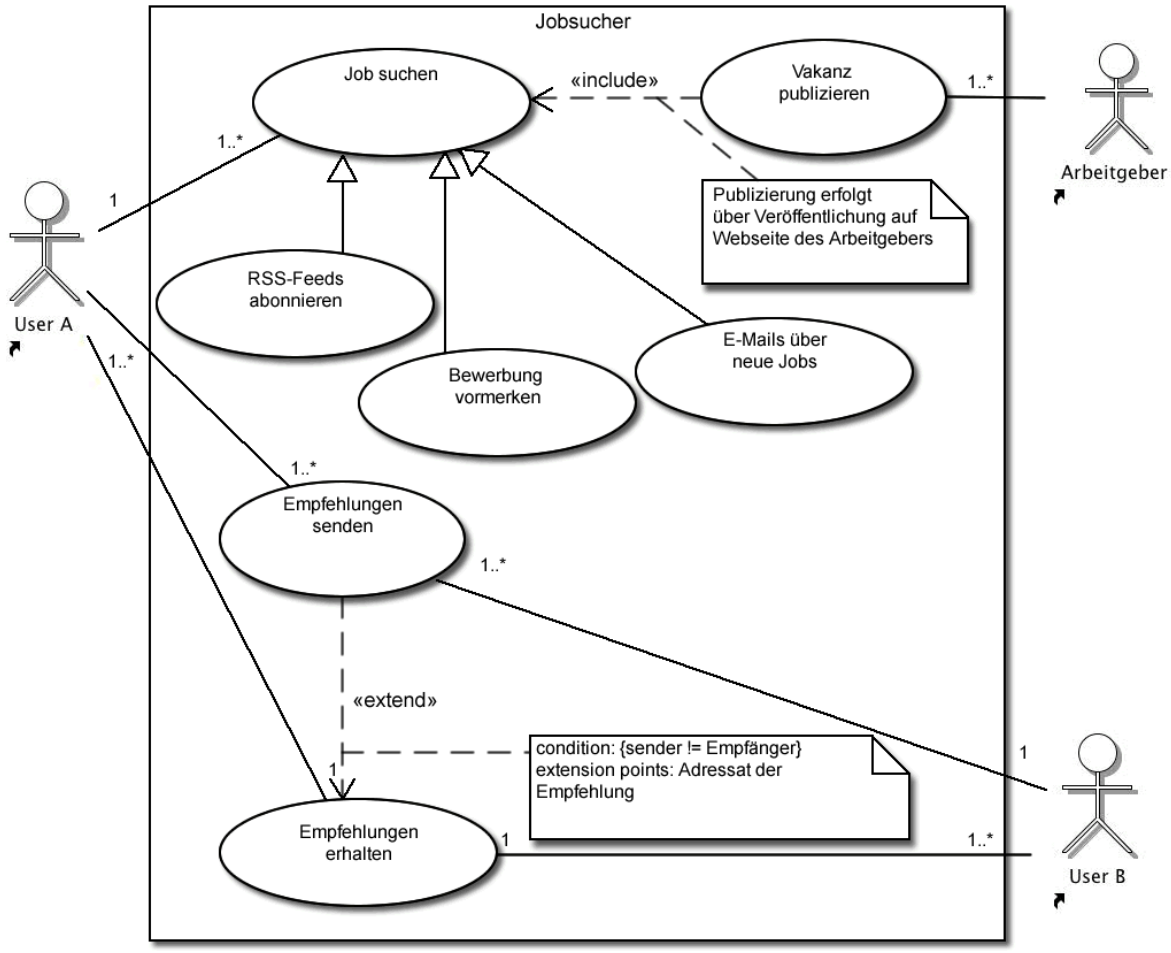


Abbildung 3.2: Use-Case-Diagramm des Jobsucher

Eine Erklärung bedarf es sicherlich beim Senden und Empfangen von prämierten Stellen. Arbeitgeber haben die Möglichkeit "Kopfgelder" auf eine Vakanz auszusetzen, die dringend besetzt werden muss. Diese Stellen werden dann, nach der Höhe des Kopfgeldes, weiter oben in der Ergebnisliste präsentiert. Vakanzen auf die ein Kopfgeld ausgesetzt ist, werden dann als prämierte Stellen bezeichnet. Durch die Kopfgelder ist also ein Relevanz-Ranking implementiert. Denn die Ergebnisse werden sonst nach dem Datum des Erfassens durch den Crawler sortiert.

Der Benutzer hat dadurch die Möglichkeit jemanden, der auf diese Stelle passt eine Empfehlung zuschicken. Tritt dieser die Stelle an, bekommt der Sender der Empfehlung das "Kopfgelder" ausgezahlt.

3.2.2 Entwicklungsprozess

Im Software-Entwicklungsprozess haben sich nach Reiner Dumke [Dum03] zwei Modellarten herausgebildet haben:

- zum einem die sequentiellen Modelle, bei denen eine relativ strenge Abfolge der Phasen zum Ausdruck gebracht wird (z.B. Wasser oder V-Modell)
- und zum anderen nichtsequentielle Modelle, diese werden auch zyklische Modelle und beinhalten mehr oder weniger Rücksprünge (Beispiele: evolutionäre Software-Entwicklung, Spiralmodell)

Jobsucher.ch ist mit Hilfe von Methoden der agilen Softwareentwicklung entstanden, bei dieser Form sind Rücksprünge zur vorhergegangen Phase besonders ausgeprägt. Bei der Softwareentwicklung gibt es immer ein klares Ziel. Ein einsetzbares System muss entstehen. Dieses System muss projektspezifischen Anforderungen entsprechen. Diese Anforderungen leiten sich direkt aus Geschäftszielen eines Unternehmens ab. Abbildung 3.3 verdeutlicht die Herausforderung bei der Softwareentwicklung. Die Geschäftsziele müssen in Anforderungen transformiert werden und diesen dann wiederum in ein System, d.h. es findet gleich ein doppelter Übersetzungsprozesse statt.

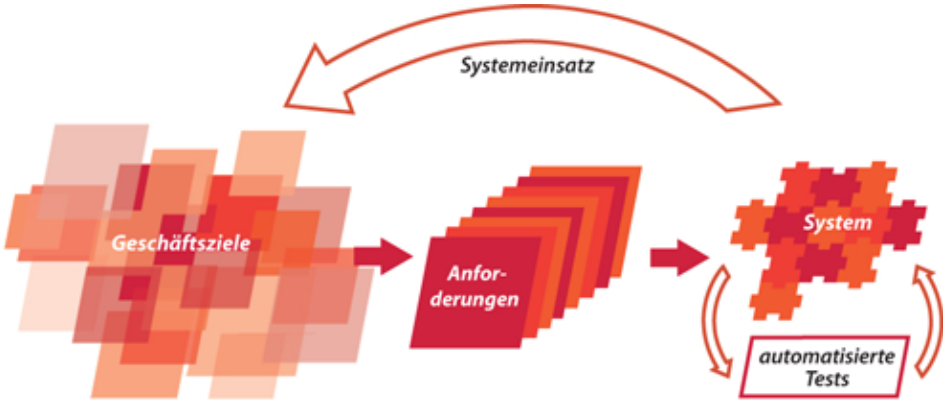


Abbildung 3.3: Agiler Zyklus über schnellen Systemeinsatz [ia09]

Zu welchem Zeitpunkt müssen nun die Ergebnisse der Übersetzungsprozesse überprüft werden? Wolf-Gideon Bleek [WGB08] schlägt dazu vor, dass der beste Weg der Überprüfung im indirekten Weg über den Systemeinsatz ist. Anforderungen werden somit nicht gegen die Geschäftsziele geprüft, sondern das erstellte System gegen die Geschäftsziele.

Diese Überprüfung sollte so früh wie möglich stattfinden. Besonders ungünstig ist eine Überprüfung am Projektende, da es dann für Korrekturen und Umstellungen zu diesem Zeitpunkt zu spät ist. Bei einer frühzeitigen Überprüfung ist es möglich in Fehlentwicklungen noch während der Projektlaufzeit einzugreifen oder diese zu umsteuern.

Nach Bleek steht hinter der agilen Softwareentwicklung ein Wertesystem, dass aus folgenden Werten besteht:

- Softwareentwicklung soll als Ganzes gesehen werden. Die Softwareentwicklungsprozesse sind für die Umsetzung von Geschäftszielen mittels Softwaresystemen verantwortlich.
- Software wird im Einsatz von Anwendern und deren Managern beurteilt und kann grundsätzlich nicht ausschließlich von Entwicklern beurteilt werden.
- Bei der Softwareentwicklung ist Rückkopplung notwendig, um zu überprüfen, ob man auf dem richtigen Weg ist, und um ggf. noch währende der Projektlaufzeit eingreifen und korrigieren zu können.

Als Voraussetzung für agile Softwareentwicklung gibt Reiner Dumke [Dum03] ein kleines Team von maximal 10 Personen und die Gewährleistung der Kommunikation untereinander an. Er beschreibt, dass der Entwicklungsprozess mit einer allgemeinen Problemstellung beginnt. Weitere Systemaspekte werden dann allerdings kommunikativ umgesetzt und weitere Anforderungen werden durch die Präsenz des Auftraggebers diskutiert. Danach teilen sich die Entwickler gegenseitig die Anforderungen und Test zu.

Nach Dumke liegt der Vorteil der agilen Softwareentwicklung im hohen Nutzereinfluss bereits zur Entwicklungszeit, denn dieser ist an der Softwareentwicklung aktiv beteiligt. Nachteilig sieht er allerdings, dass sich die personengebundenen Systemkenntnisse negativ auf die Wartbarkeit auswirken.

Das Entwicklungsteam von jobsucher.ch bestand aus drei aktiven Personen, einem Webdesigner und zwei Entwicklern. Vom Webdesigner wurde die komplette Seite in Form eines sogenannten Mock-ups[15] geliefert. Ein Entwickler war für die Realisierung des Crawlers und die damit verbundene Service-API zur Kommunikation verantwortlich. Der zweite Entwickler entwickelte die neuen Features im Frontend des Jobsuchers und baute die neue Service-API in den bestehenden Jobsucher ein.

Bei der Entwicklung wurde eine Mischform aus den beiden agilen Methoden Scrum und Feature Driven Development verwendet. Auf eine genaue Beschreibung der beiden Methoden wird in dieser Arbeit verzichtet, sondern darauf eingegangen, welche Punkte daraus zur Anwendung gekommen sind.

Zur Abstimmung des Teams setzt man bei Scrum auf ein tägliches Treffen, dieses sogenannte Daily Scrum wurde auf ein wöchentliches Treffen reduziert. Bei diesem Treffen beantworten die Beteiligten, die folgenden drei Fragen:

1. Was habe ich seit dem letzten Weekly Scrum getan?
2. Was werde ich bis zum nächsten Weekly Scrum tun?
3. Was behindert mich?

[15] Mock-up (engl.: “Attrappe”) ist ein Vorführmodell, dass die kompletten Funktionen der Webseite simuliert.

Feature Driven Development besteht aus fünf Teilprozessen, die in Abbildung 3.4 dargestellt werden und bei der Entwicklung des Jobsuchers in folgender Form zur Anwendung kamen:

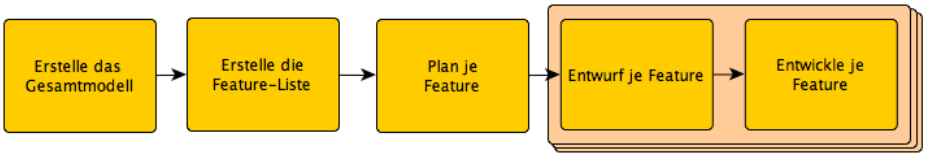


Abbildung 3.4: Schema des Feature Driven Development - Prozess

1. **Erstelle das Gesamtmodell**

   Wie schon erwähnt wurde die gesamte Webapplikation in Form eines Mock-ups realisiert, die alle Funktionen simuliert.

2. **Erstelle die Feature-Liste**

   Dieses Mock-up diente also gleichzeitig als eine Feature-Liste. Denn es simuliert die einzelnen Feature der Webseite. Neue Feature-Requests wurden im Bug-Tracker Mantis[16] erfasst und später in das Mock-up übernommen.

3. **Plane je Feature**

   Die einzelnen neuen Features wurden nur priorisiert, wenn Sie für das Deployment wichtig waren. Ansonsten wurden sie beim Refactoring des bestehenden Quellcodes eingebaut.

4. **Entwurf je Feature**

   Durch die flache Struktur des Entwicklungsteam gab es auch keinen Chefprogrammier der die Features plante. Dieser Task wurde vom Entwickler selbst übernommen.

5. **Entwickle je Feature**

   Nach dem er das entsprechende Feature geplant hat, begann er auch die Implementierung der neuen Funktion.

Mit der Anwendung von agilen Methoden rückt auch die Qualitätssicherung in den Vordergrund, das wird in Abbildung 3.3 angedeutet. Durch die kleinen Entwicklungsschritte und die daraus resultierenden kurzen Releasezeiten, muss durch geeignete Verfahren sichergestellt werden, dass Fehler gefunden und beseitigt werden. Dazu müssen geeignete Testverfahren gefunden werden, wie z.B. das Unit-Testing.

[16] http://www.mantisbt.org/ - 02.08.2009

Der klassische Unit Test besteht aus einer Menge von Testfällen. Diese werden in Form einer Methode repräsentiert und testeten auch eine Methode. Eine überschaubare Anzahl an Testfällen wird in einer Testklasse gruppiert, vgl. [Mef06].

3.2.3 Maven

Wünschenswert ist es diese Tests beim Build der Applikation automatisch durchzuführen. Die Aufgabe übernimmt Maven, doch es kann natürlich noch viel mehr. Bei Maven[17] handelt es sich um ein Projektmanagement-Tool, dass bei der Erstellung des Jobsucher zum Einsatz kam. Das Wort "Maven" stammt ursprünglich aus dem Hebräischen bzw. Jiddischen und bedeutet grob übersetzt "Verständnis" oder "Sammler von Wissen". Apache Maven definiert das Projekt, wie folgt:

> "Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information."

Maven verfolgt nach Vincent Massol [VM06] und Sebastian Karb [Kar08] die folgenden Ziele:

- Maven hilft bei der Erstellung von Buildprozessen, welcher mit zahlreichen Erweiterungen und Plugins erweitert werden kann.
- Maven führt ein einheitliches Project Object Model (POM) ein, damit wird das komplette Projekt und seine Handhabung in einer XML-Datei beschrieben.
- Alle Projekte besitzen einen einheitlichen Lebenszyklus. Dieser wird durch die Projektschablonen (Archetypes) beschrieben, die das Projekt in verschiedene Phasen und deren Ziele unterteilt ist.
- Maven bietet ein umfassendes Abhängigkeitsmanagement. Repositories vereinfachen dabei die Wiederverwendung von Bibliotheken.
- Maven ist plattformunabhängig, da es in Java geschrieben wurde.
- Auch die Erstellung von Dokumentationen, Berichten, Webseiten und Tests von Projekten kann einfach integriert werden.

Maven verfolgt dabei das Grundprinzip "Convention Over Configuration" (Konvention vor Konfiguration). Dadurch wird der Aufwand für die Konfiguration auf ein Minimum begrenzt. Das wird über einen hohen Grad an Vorgaben und Richtlinien sichergestellt. Das ist wiederum ein sehr wichtiger Aspekt in der agilen Softwareentwicklung. Der Benutzer verschwändet so nicht viel Zeit für die Konfiguration oder das Aufsetzen der Arbeitsumgebung, sondern kann ein neues Projekt direkt erstellen oder ein bestehendes Projekt bearbeiten. Ausserdem ermöglicht dieser Ansatz eine unternehmensweite Basis für Projekte zu schaffen, mit der alle Beteiligten gleich arbeiten.

[17] http://maven.apache.org/ - 03.08.2009

3.3 Arbeitsweise des Jobsuchers

3.3.1 Aufbau des User-Interface

Zur Eingabe einer Suchfrage besitzt der Jobucher das in Abbildung 3.5 darstellte User-Interface. Bei der Betrachtung findet man die im Kapitel 2.2.2 vorgestellten Punkte zur Gestaltung wieder. Es ist zu erkennen, dass es sehr einfach gestaltet ist und nur wenige Möglichkeiten der Modifikation der Suchanfragen bietet. In der Regel wird nur nach einer Berufsbezeichnung und einem Ort gesucht.



Abbildung 3.5: Suchfunktionen des Jobsucher

Um die Suchanfrage weiter einzugrenzen hat man die Möglichkeiten die Eingabemaske zu erweitern und die Suche über unternehmensspezifische Merkmale weiter zu verfeinern. Diese Merkmale sind die Unternehmensgröße, der Sektor im dem das Unternehmen tätig ist, der Markt in dem es agiert oder Rubriken, der das Unternehmen zugeordnet ist. Ausserdem hat man die Möglichkeit sich nur prämierte Jobs anzeigen zu lassen oder die Stellen auszublenden, die durch eine Personalagentur vermittelt werden.

3.3.2 Aufbau der Jobsucher-Ontologie

Bevor nun ein technischer Einblick darüber erfolgt, was mit einer Suchanfrage passiert, ist es nötig den Aufbau der Ontologie zu erklären. Im Kapitel 2.3.4 wurde RDF als formale Sprache zur Beschreibung einer Ontologie vorstellt. Es ist aber auch möglich eine relationale Datenbank zu benutzen, um ein Tripel “Subjekt-Prädikat-Objekt” zubeschreiben. In diesem Ansatz wird das Tripel aus zwei Konten in der Datenbank und der Kante, die diese beiden Knoten verbindet, gebildet.

Bei einer Betrachtung Abbildung 3.6, die das zu Grunde liegenden Datenbankschema darstellt, fallen die Tabellen “tbfulljob”, “tbsynoym” und “tbnosynoym” in das Auge. In der Tabelle “tbfulljob” werden die Berufsbezeichnung, wie z.B “CEO”, gespeichert. Die

entsprechenden Synonyme, wie Konzernleiter, Geschäftsführer und Managing Partner usw, werden in der Tabelle “tbsynoym” und die Antonyme in der Tabelle “tbnosynoym” gespeichert.

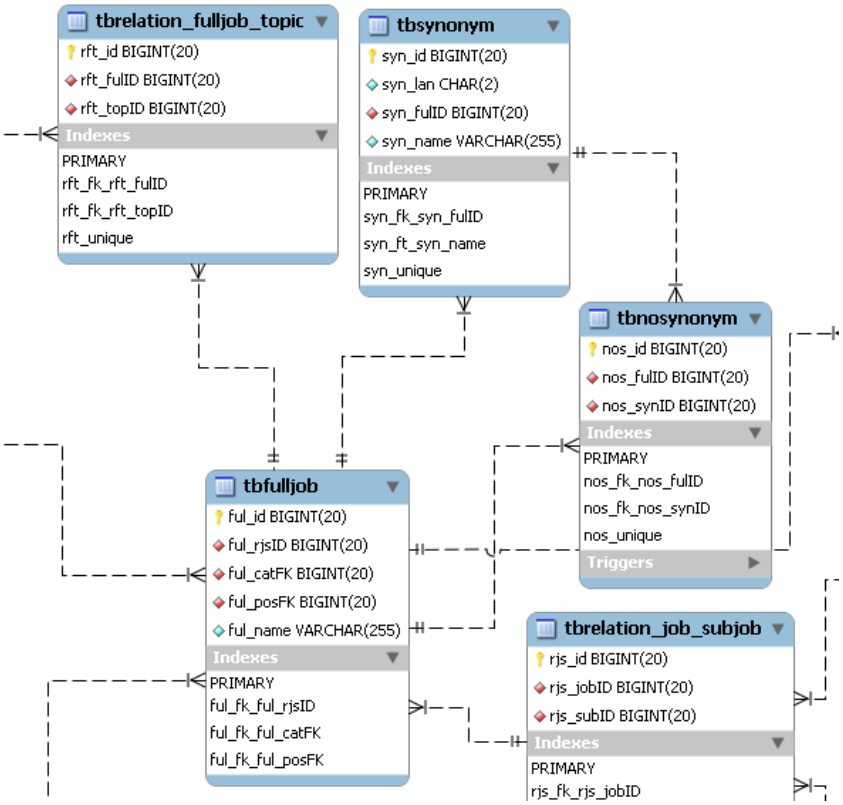


Abbildung 3.6: Ausschnitt des Datenbank Schema der Ontologie

Die Kante, die beide Tabellen mit einander verbindet, spiegelt das Prädikat wieder. Die Beziehung lautet “bfulljob hat das Synonym tbsynoym”, die Verbindung zwischen den beiden Tabellen erfolgt über das Attribute “syn_fulID” von “tbsynoym” in dem die “ful_ID” von “tbfulljob” gespeichert wird. Das selbe Prinzip findet man auch bei den Antonymen wieder. Damit ist eine deutliche Analogie zum RDF-Graphen zu erkennen.

Vorteile	**Nachteile**
RDF-Dokumente sind portabel	Verlängerung der Entwicklungszeit
RDF-Dokumente sind visualisierbar	Parsen von RDF-Dokumente geht auf Kosten der Performance

Tabelle 3.1: Vor- und Nachteile von RDF gegenüber einer relationalen Datenbank

Die Tabelle 3.1 zeigt die Vor- und Nachteile einer Implementierung mit RDF gegenüber einer relationalen Datenbank. RDF-Dokumente sind durch ihre XML-basierte Struktur portabler. Denn es gibt für jede Programmiersprache Frameworks zum Parsen von XML-Dateien. In der Programmiersprache Java kann man dafür beispielsweise die

Frameworks JAXP[18] oder JDOM[19] verwenden. Außerdem gibt es eine Reihe von Tools zur Visualisierung von RDF-Dokumenten, wie z.B. IsaViz[20] oder Smore[21].

Allerdings ist die Einarbeitungszeit für den Entwickler durch RDF größer. Denn er muss sich zuerst mit der Syntax für RDF-Dokumente vertraut machen und sich dann in die entsprechenden Frameworks einarbeiten. SQL als Anfragesprache für Datenbanken ist den meisten Entwicklern jedoch bekannt und erfordert nur wenig Aufwand zum Verstehen des Datenbankschemas. Außerdem geht das Parsen von XML-Dateien auf Kosten der Performance der Anwendung.

Abschießend wird drauf hingewiesen, dass Ansätze zum persistieren von RDF-Daten existieren, auf die zu Gunsten der Entwicklungszeit verzichtet wurde. Dazu wird auf die Arbeiten von Alexander Haupt [Hau04] und Christian Lehmann [Leh06] verwiesen.

3.3.3 Funktionsweise des Jobsuchers

Um die technische Funktionsweise des Jobsucher zu erklären, werden die aus Kapitel 2 bekannten vier Methoden eines IRS und ihre Anwendung beschrieben. Die Architektur der Suchmaschine ähnelt Abbildung 2.4 mit der Ausnahme, dass kein automatischen Crawler existiert.

Beschaffen und Erfassen der Daten

Das Beschaffen und Erfassen von Daten übernimmt ein halb-automatischer Web-Crawler, der Webseiten in vordefinierten Zeitintervallen besucht und die Daten der Stellenanzeigen aus den Webseiten extrahiert. Dazu existieren manuell erfasste Profile, in den die relevanten Informationen mit Hilfe von XPath[22]-Anweisungen beschrieben werden. Die durch XPath adressieren Stellen in der HTML-Seite dienen dem Crawler zur Navigation auf der Seite oder sie adressieren die Informationen und werden dann entsprechenden Datenfelder zugeordnet.

Indexieren

Zur Indexierung wird das Lucene[23]-Framework eingesetzt. Es handelt sich dabei um ein Suchmaschinen-Framework und stellt Klassen und Methoden bereit, um Dokumente zu analysieren, zu indexieren und zu speichern. Alles in Textform wird verarbeitet, wie z.B. ASCII-Dateien, Datenbankfelder oder E-Mails. Um Inhalte in Formaten wie HTML, PDF, Microsoft Word oder OpenOffice zu verarbeiten, werden Filter benötigt um die textuellen Informationen zu extrahieren.

[18] https://jaxp.dev.java.net/ - 09.08.09
[19] http://jdom.org - 09.08.09
[20] http://www.w3.org/2001/11/IsaViz/ - 09.08.09
[21] http://www.mindswap.org/2005/SMORE/ - 09.08.09
[22] http://www.w3schools.com/XPath/ - 21.06.09
[23] http://lucene.apache.org/ - 03.08.09

Die Textinhalte werden an den so genannten Analyzer übergeben. Die Daten werden dann nach fest definierten Regeln zerlegt und in einer einheitlichen Weise für den Index aufgearbeitet. Nach der Analyse erfolgt die eigentlich Indexierung und das Abspeichern des aufbereiteten Suchraums. Dazu werden die folgenden Indexfelder unterschieden:

- "indexed" gibt an, ob ein Suchfeld durchsuchbar sein soll
- "tokenized" gibt an, ob der Eingabestream durch einen Analyzer normalisiert wird
- "stored" gibt an, ob das Feld in seiner Orginalform im Index abgespeichert wird

Das Lucene-Framework unterstützt Term- und Phrasensuche, Boolesche Operatoren, Wildcard-Suche, Range-Queries sowie Fuzzy- und Proximity-Suche. Vor dem Suchvorgang muss die Anfrage erst in einem abstrakten Anfrageobjekt überführt werden. Dafür kann man den Query-Parser verwenden.

Der integrierte Query Parser lehnt sich in seiner Syntax stark an Googles Anfragesprache an. Das erzeugte Anfrageobjekt wird anschließend an einen Index-Searcher übergeben. Dieser übernimmt dann die Suche im technischen Sinn.

Speichern

Das Persistieren, der durch den Crawler gesammelten Vakanzen, wird dann mit Hibernate[24] gehandhabt. Damit ist es möglich den Zustand eines Objektes in einer relationalen Datenbank zu speichern und umgekehrt aus den Datensätzen wieder Objekte zu erzeugen, diesen Prozess bezeichnet man als objekt-relationales Mapping. Zudem befreit Hibernate den Entwickler bei der Programmierung von SQL-Abfragen und man kann die verwendete Datenbank einfach austauschen.

Recherchieren

Da es sich beim Jobsucher um eine Web-Applikation handelt greift man natürlich auch auf Design Pattern einer klassischen Web-Anwendungen zurück. Diese sind z.B. das MVC-Konzept und die Dependency Injection. Nach Hemraja [Hem07] findet man das MVC-Konzept heute in dem meisten Java-basierten Webframeworks wieder. Im Jobsucher wurden Spring[25] in Verbindung mit Hibernate für die Realisierung werdet.

Bei Spring handelt es sich um ein Framework, dass das Ziel verfolgt die Entwicklung mit Java zu vereinfachen. Es bietet mit einem breiten Spektrum an Funktionalität eine Lösung zur Entwicklung von Anwendung- und Geschäftslogik. Spring basiert auf Prinzipien, die in dem Buch "Expert One-On-One J2EE Design und Development" von Rod Johnson vorgestellt werden:

- **Dependency Injection** minimiert alle Abhängigkeiten zwischen den Objekten. Dazu existieren zwei Injection-Arten. Man kann man die Argumente entweder über den Konstruktor oder die Setter-Methoden einer Klasse übergeben.

[24] http://www.hibernate.org/ - 21.06.08
[25] http://www.springsource.org/ - 21.06.08

- **Aspekt-orientierte Programmierung** sorgt für eine Isolierung von technischen Aspekten, wie z.B. Transaktionen oder Sicherheit. Durch die Trennung der logischen Aspekte wird der Code der Anwendung davon freigehalten.

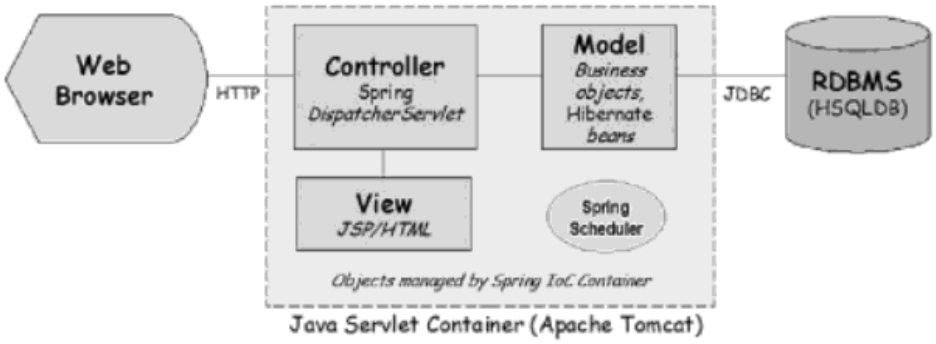


Abbildung 3.7: High-Level-Architektur für Time Expression

In Abbildung 3.7 wird nun das Zusammenspiel von Spring und Hibernate gezeigt und wie das MVC-Konzept durch diese beiden Frameworks realisiert wird. Das Spring Dispatcher Servlet dient zur Steuerung der gesamten Web-Applikation. Es leitet die Anfrage an eine entsprechende Klasse weiter, die die Anwendungslogik enthält. Diese hat die Möglichkeit über entsprechende Service-Klassen Anfragen an das Datenmodell zustellen und in ihr wird die weitere Verarbeitung dieser Daten organisiert. Die Ergebnisse der Verarbeitung werden dann an den View übergeben.

Vorteile	**Nachteile**
alle Komponenten sind unabhängig voneinander und austauschbar.	strikte Konventionen an die sich Entwickler halten müssen
Zwang für den Entwickler sauber strukturierten Quellcode zu schreiben	Kosten der Performance.
Arbeitsteilung gemäß Fähigkeiten	
Alle Komponenten sind wiederverwendbar	

Tabelle 3.2: Vor- und Nachteile des MVC-Konzept [Geh08]

Die Vor- und Nachteile des MVC-Konzepts werden in Tabelle 3.2 dargestellt. Alle Komponente sich unabhängig von einander, austauschbar und wiederverwendbar. Der Entwickler wird zu sauber strukturierten Quellcode gezwungen. Der bedeutsamste Vorteil liegt darin, dass sich Aufgaben geteilt werden können. Der Webdesigner kann dadurch Views erstellen, ohne Einbild in die Implementierung der Anwendungslogik zu haben. Er muss lediglich wissen, welche Werte an den View übergeben werden.

Nachteile sind jedoch die strikten Konventionen an die sich die Programmier halten müssen. Außerdem geht der Einsatz des MVC-Konzeptes auf Kosten der Performance.

Nachdem nun die Eingabemaske aus Abbildung 3.5 ausgefüllt und das Suchformular abschickt wurde, bekommt der Server einen HTTP-Request. Dieser wird nun vom Spring Dispatcher Servlet an die Klasse "ResultController" weitergeleitet, die die Anwendungslogik enthält. Diese übernimmt die weitere Bearbeitung des Request und damit der Suchanfrage.

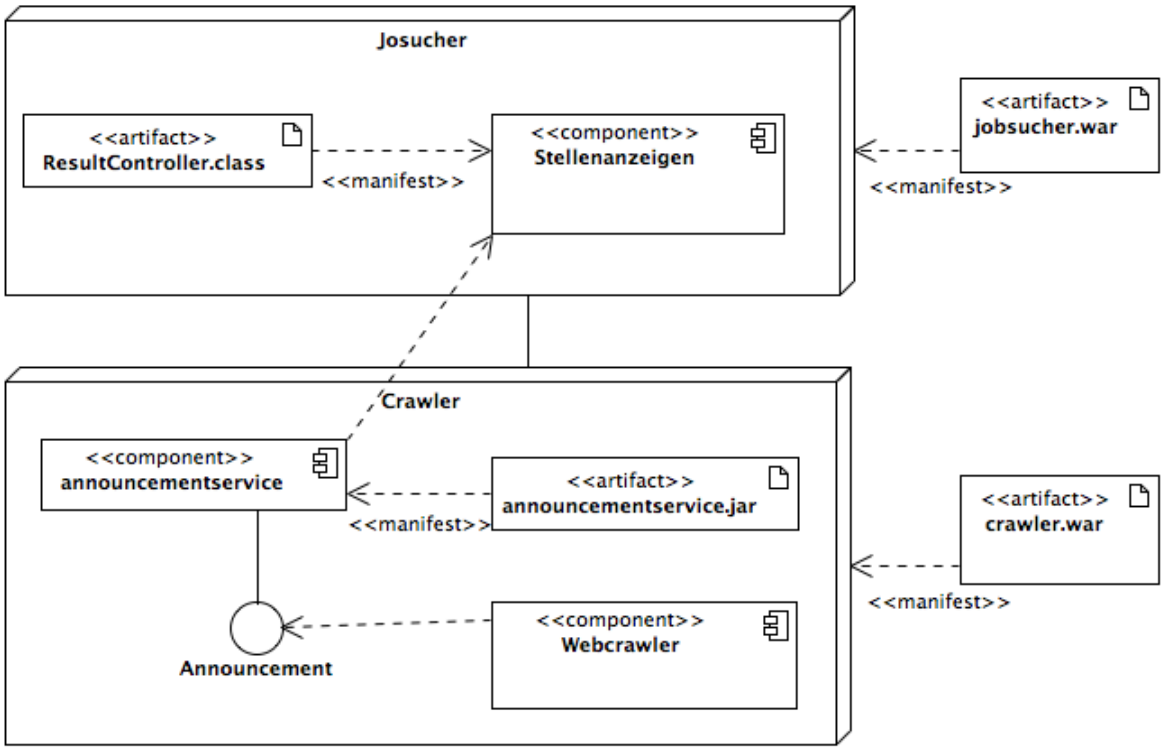


Abbildung 3.8: Verteilungsdiagramm Jobsucher und Crawler

Der "ResultController" ruft nun den "Announcementservice" mit dem entsprechenden Backing-Objekt[26] des Suchformulars auf. Vom "Announcementservice" wird ein Hashwert des Suchformular generiert. Mit dem zusammen wird das Suchformular in die Datenbank gespeichert.

Mit dem Hashwert des Suchformulars wird nun im "Searchservice", der im "Announcementservice" enthalten ist, ein Suchstring für MySQL oder Lucene generiert. Das wird durch eine API realisiert, die vom Aufbau der Critieria API von Hibernate ähnelt. Bei der Generierung des Suchstrings wird auf den "Ontologyservice", der ebenfalls im "Announcementservice" enthalten ist, zugegriffen. Dieser ermittelt anhand der Stellenbezeichnung eine "full_ID" und ordnet der "full_ID" die entsprechenden Synonyme zu, die dann in den Suchstring einbaut werden.

[26] Ein Backing-Objekt ist ein Java-Objekt im dem alle Daten eines Formulars gespeichert werden.

Wenn keine “full_ID” zugeordnet werden kann, kommt es zu einem Volltext-Matching. Die Ergebnisse werden nach dieser Bearbeitung an die View weitergeleitet und durch eine JSP werden dann die Suchergebnis dargestellt.

Die Kommunikation zwischen Crawler und Jobsucher erfolgt über den “Announcementsservice”, wie Abbildung 3.8 zeigt. Durch diesen Service werden beide Systeme mit eine gemeinsame Datenbank verbunden. Als Datenbankmanagementsystem kommt hierbei MySQL zum Einsatz. Hibernate stellt eine Verbindung zwischen Announcementservice und Datenbankserver her. Eine mögliche alternative Implementierung Mithilfe von Webservice wurde von Martin Sturm [Stu08] schon im Rahmen seiner Arbeit “Design und Implementierung des Onlinesystems Bewerbungsorganisator” besprochen. Aus diesem Grund wird in dieser Arbeit darauf verzichtet.

Kapitel 4

Auswertung der Ergebnisse

4.1 Auswertung der Suchergebnisse

Für die Auswertung der Suchergebnisse werden die aus Kapitel 2.1.4 bekannten Metriken *Precision* (Genauigkeit) und *Recall* (Vollständigkeit) nun für eine Suchanfrage berechnet.

4.1.1 Bestimmung der Precision

In der Praxis wird die Precision bestimmt, in dem der Benutzer alle selektierten DE bearbeitet. Dann entscheidet er, ob sie seiner Meinung nach für die Suchanfrage relevant sind. Damit ist die Anzahl der gefundenen relevanten Treffer (Größe a) gegeben. Die Gesamtanzahl der gefundenen Dokumente ist ohnehin bekannt. Diese Werte werden dann in die Formel 2.3 einsetzt.

Als Suchanfrage wird der Begriff "CEO" und der Stand der Anzeigendatenbank vom 23.06.09 verwendet. Die Suchanfrage nach diesem Begriff führte zu 72 gefundenen Stellen. Von diesen 72 Stellen werden die folgenden fünf Stellen als nicht relevant bewertet:

- Direkt zur Navigation,
- Stv. Geschäftsführer mit Übernahmeoption,
- Professur/Direktor,
- Chief Finacial Officer (CFO) Banking,
- Stv. Geschäftsführer/in.

So ergibt sich für die Größe a (gefundene relevante Dokumente) ein Wert von 67 und für die Größe $a + b$ (Gesamtanzahl der gefundenen Dokumente) ein Wert von 72. Diese Werte werden nun in Formel 2.3 eingesetzt und das Ergebnis berechnet:

$$Precision = \frac{a}{(a+b)} = \frac{67}{72} \approx 0,93056 \tag{4.1}$$

Auf diese Weise wird für die Genauigkeit auf die Suchanfrage “CEO” ein Wert von 93,056% berechnet.

4.1.2 Bestimmung der Recall

Die Bestimmung der Vollständigkeit ist nicht so trivial, denn die Größe c (relevante Dokumente, die nicht gefunden wurden) in Formel 2.1 lässt sich nicht so einfach messen. Um den Wert der Größe c zu ermitteln, müssen alle 39926 Stellenanzeigen auf die Relevanz für eine Suchanfrage manuell überprüft werden. Jeder kann sich vorstellen, dass das sehr mühsam ist und viel Zeit in Anspruch nimmt.

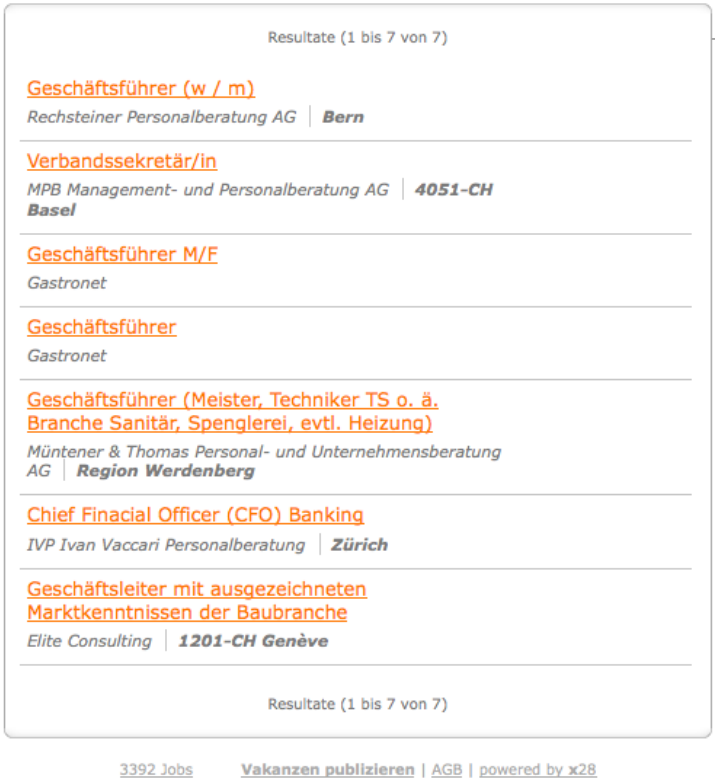


Abbildung 4.1: Ergebnisse der Stichprobe von 3392 Anzeigen

Da es sich bei einer Jobsuchmaschine um eine sehr begrenzte Thematik handelt, kann nach Gaus [Gau05] das Stichprobenverfahren verwendet werden. Dazu wird eine bestimmte Anzahl an Einträgen zufällig ausgewählt, z.B. 10% aller in der Datenbank befindlichen Anzeigen und diese werden auf die Relevanz für eine bestimmte Suchanfrage überprüft.

Als Suchanfrage wird auch hierbei der “CEO” und die Datenbank vom 23.06.09 verwendet. Aus den 39926 aktiven Anzeigen der Datenbank werden 3992 zufällig ausgesucht und auf die Relevanz zur Suchanfrage überprüft. Die gefundenen Stellen aus dieser Stichprobe werden in Abbildung 4.1 dargestellt. Die manuelle Überprüfung der Stichprobe ergibt, dass folgende Stellen als relevant bewertet werden:

- Geschäftsführer Sportbekleidungsmarke
- Geschäftsführer
- Geschäftsführer (Meister, Techniker TS o.ä. Branche Sanitär, Spenglerei, evtl. Heizung)
- Geschäftsführer (w / m)
- Geschäftsführer / in Geotechnik (Feststelle, 100%)
- Geschäftsführer M/F
- Geschäftsleiter mit ausgezeichneten Marktkenntnissen der Baubranche
- Leiter/in Generalagentur
- Verbandssekretär/in

Daraus ergibt sich für die Anzahl der relevanten Stellen in der Stichprobe ein Wert von 9 Stellen. Die geschätzte Anzahl relevanter Dokumente $a + c$ lässt sich nun mit der folgenden Formel bestimmen:

$$a + c = \frac{\text{Anz. d. relev. DE i. d. Strichprobe} * \text{Anz. d. DE i. d. Dokumentation}}{\text{Anz. d.DE in der Stichprobe}} \tag{4.2}$$

Als Werte werden nun die Anzahl der relevanten Stellen in der Stichprobe mit dem Wert 9, die Gesamtanzahl der Stellen mit dem Wert 39926 und die Anzahl der Stichprobe mit den Wert 3992 in die Formel 4.2 eingesetzt.

$$a + c = \frac{9 * 39926}{3992} \approx 89,991 \tag{4.3}$$

Damit sind nun alle Werte der Variablen aus Formel 2.1 bekannt und werden in sie eingesetzt.

$$Recall = \frac{a}{(a + c)} = \frac{67}{89,991} \approx 0,74451 \tag{4.4}$$

Die Vollständigkeit der Suchanfrage “CEO” liegt damit bei 74,451%.

4.1.3 Vergleich der Precision mit anderen Suchmaschinen

Das Vorgehen aus Kapitel 4.1.1 wird benutzt, um jobsucher.ch mit anderen Jobsuchmaschinen zu vergleichen. Die Ergebnisse der Genauigkeitsmessung sind in Abbildung 4.2 dargestellt und wurden am 06.01.09 erhoben.

Klar zu erkennen ist, dass der Jobsucher verglichen mit anderen schweizer Jobsuchmaschinen mit Abstand die höchste Genauigkeit bei der Suchanfrage “CEO” besitzt. Dies ist auf die Verwendung der Ontologie zurückzuführen.

Name / URL	Gesamt-anzahl gefundener Vakanzen	Resultate bei Suche nach „CEO" in der Schweiz	Semantisch passende Resultate	Precision (= % der gefundenen Jobs, die relevant sind)
jobsucher www.jobsucher.ch	32'289	68	61	90%
jobseek.ch www.jobseek.ch	30'115	38	1	2,6%
jobsuchmaschine.ch www.jobsuchmaschine.ch	12'184	42	3	7,2%
DirectJobs www.direct-jobs.ch	4'264	63	4	6,3%
jobrapido www.jobrapido.ch	89'568	123	9	7,3%
careerjet www.careerjet.ch	52'951	262	5	2%

Abbildung 4.2: Vergleich mit anderen Jobsuchmachinen

Das wird einer der Gründe gewesen sein, dass jobsucher.ch von anthrazit, einem renommierten schweizer Internet-Magazin, am 11.02.2009 unter die besten 200 Webseiten der Schweiz 2009 gewählt wurde. Seit diesem Tag schmückt diese Auszeichnung die Suchmaschine, vgl. [Ant09].

Eine Vergleich hinsichtlich der Vollständigkeit kann nicht erfolgen. Denn die Datenbanken der anderen Suchmaschinen sind nicht öffentlich zugänglich.

4.2 Bewertung der Ontologie

Abschließend werden noch Punkte genannt, die das Ergebnis einer direkten Kommunikation mit ZHAW (Zürcher Hochschule für Angewandte Wissenschaften) Winterthur sind. Diese fand im Mai 2009 statt und führte zu dem folgenden Ergebnissen:

- Innerhalb einzelner Konzepte ist eine Inferenz von allgemein hierarchischer Oberbegriff zum spezifischen Unterbegriffen möglich.
- Die Struktur der Ontologie erscheint (auch aufgrund der gegebenen notwendigen Abbildung der realen Welt) als nicht durchgängig systematisch.
- Die Suche ist noch nicht sprachübergreifend, d.h. mit einer Anfrage auf deutsch werden noch nicht die entsprechenden französischen oder italienischen Texte gefunden (z.B. findet eine Anfrage nach “Maurer” nicht Stellenangebote für einen “maçon”); französische, italienische und englische Einträge sind zwar teilweise bereits in der Ontologie vorhanden, aber nicht miteinander verknüpft.

Kapitel 5

Zusammenfassung und Ausblick

5.1 Zusammenfassung

Diese Arbeit verfolgt zwei Ziele. Das Erste ist es, Analogien zu einem klassischem IRS aufzuzeigen. Bei der Jobsuchmaschine jobsucher.ch sind natürlich klassische Methoden eines IRS wiederzufinden. Das Wiederauffinden von Vakanzen wird durch die folgenden vier Prozesse eines IRS ermöglicht:

1. **Beschaffen und Erfassen**

   Diese Aufgabe übernimmt ein halb-automatischer Crawler, der Firmen-Webseiten in bestimmten Zeitintervallen besucht und Vakanzen mit Hilfe von manuell erfassten Profilen extrahiert.

2. **Indexieren**

   Die vom Crawler erfassten Informationen über freien Stellen werden nach dem Erfassen indexiert, um ein schnelles Auffinden zu ermöglichen.

3. **Speichern**

   Die Stellen werden im nächsten Schritt in einer Datenbank persistiert.

4. **Recherchieren**

   Der letzte Schritt ist das gezielte Wiederauffinden von pertinenten Stellen, dass über ein Web-Interface erfolgt, im dem Gestaltungsmerkmale einer normalen Websuchmaschine wieder zu finden sind.

Das zweite Ziel dieser Arbeit ist es aufzuzeigen, dass durch die Verwendung einer Ontologie die Qualität der Suchergebnisse verbessert wird. Die Arbeit zeigt, dass die verschiedenen Bezeichnungen für einen Beruf dazu benutzt werden, die Anzahl an gefundenen relevanten Stellen zu erhöhen. Dazu werden Synonyme verwendet. Über Antonyme werden Berufe ausschlossen, die nichts mit einer Vakanz zu tun haben. Mit deren Hilfe wird die Anzahl an gefundenen nicht relevanten Stellen reduziert. Die Beziehungen zwischen

den Berufsbezeichnungen werden durch eine Ontologie abgebildet. Diese befindet sich in einer Datenbank und ähnelt einem RDF-Graphen. Durch deren Verwendung wird die Genauigkeit der Suchmaschine verbessert, wie man der Auswertung in Kapitel 4 entnehmen kann.

5.2 Ausblick

Abschießend ist noch zusagen, dass der Jobsucher in den folgenden Punkten verbessert werden kann:

- Gerade in einem mehrsprachigen Land, wie der Schweiz, ist eine sprachübergreifende Suche erforderlich. Dazu muss die Ontologie in linguistischer Sicht verbessert werden.
- Die Arbeiten von Alexander Haupt [Hau04] und Christian Lehmann [Leh06] beschreiben Ansätze, um RDF-Daten in einer Datenbank zu speichern und Daten aus einer relationalen Datenbank in RDF-Dokumente zu mappen. Diese Arbeiten geben Grund zur Überlegung, ob es nicht sinnvoll ist die Ontologie mit Hilfe von RDF zu beschreiben, um sich so an Standards zuhalten, die durch das W3C vorgeschlagen werden.
  Eine Kooperationen mit einer Jobsuchmaschinen existiert bereits. Diese nutzt die Job-Ontologie. Durch eine Standardisierung der Ontologie hätte man aber die Möglichkeit, weitere Kooperationen mit anderen Suchmaschinen zu vereinfachen.
- Die Auswertung der Vollständigkeit und Genauigkeit haben gezeigt, dass die Suche zwar Ergebnisse mit einer sehr hohen Genauigkeit geliefert werden, aber auch viele relevante Stellen verloren gehen. Das erfordert eine Überprüfung, des zu Grunde liegenden Matching-Algorithmus.
- Zudem ist eine Untersuchung der Vollständigkeit und der Genauigkeit auch für andere Job-Bezeichnungen als den “CEO” erforderlich.

Literaturverzeichnis

[Ant09] Anthrazit: *http://www.anthrazit.org/appl/download.php?apsid=37a4334f&objid=94475494&id=1339920827&nr=1.* letzter Besuch - 17.08.09, 2009.

[BSC97] Ben Shneiderman, D. B.; Croft, W. B.: Clarifying search: A user-interface framework for text searches. D-Lib Magazine, 1997.

[CDM08] Christopher D. Manning, H. S., P. R.: *Introduction to Information Retrieval.* Cambridge University Press, 2008.

[Dum03] Dumke, R.: *Software Engineering.* vieweg, 2003.

[Gau05] Gaus, W.: *Dokumentations- und Ordnungslehre.* Springer, 2005.

[Geh08] Gehnke, M.: *Das Model-View-Controller-Konzept in Webanwendungen am Beispiel des Zend-Frameworks.* Hochschule Wismar, Wirtschaftliche Fakultät, 2008.

[Gra07] Graupmann, J.: *Semantische Suche - Intelligente Suchmaschinen durch innovative und zukunftsweisende Konzepte und Technologien.* VDM Verlag Dr. Müller, 2007.

[Hau04] Haupt, A.: *Persistentes Speichern von RDF-Daten.* Humbold Universität Berlin, 2004.

[Hem07] Hemrajani, A.: *Agile Java-Entwicklung mit Spring, Hibernate & Eclipse.* mitp, 2007.

[ia09] agile i.: *http://www.it-agile.de/wasistagilesoftwareentwicklung.html.* letzter Besuch - 09.08.09, 2009.

[Kar08] Karb, S.: *Projektmanagement mit Maven.* Fachhochschule Wedel, 12 2008.

[Leh06] Lehmann, C.: *R2D2 – a Language for Mapping between Relational Data and RDF-based Ontologies.* Universität Leipzig, 2006.

[Lew05] Lewandowski, D.: *Web Information Retrieval - Technologien zur Informationssuche im Internet.* DGI, 2005.

[Mef06] Meffert, K.: *JUnit Profi-Tipps - Software erfolgreich testen.* entwickler.press, 2006.

[Mün09] Münster, U.: *http://santana.uni-muenster.de/Linguistik/user/steiner/ semindex/intro.html*. letzter Besuch - 26.06.09, 2009.

[PH08] Pacal Hitzler, S. R. Y. S., M. K.: *Semantic Web*. Springer, 2008.

[Poe06] Poetzsch, E.: *Information Retrieval - Einführung in Grundlagen und Methoden*. E. Poetzsch, 2006.

[RD03] Reiner Dumke, C. W., M. L.: *Web Engineering*. Pearson Studium, 2003.

[Sto07] Stock, W. G.: *Information Retrieval - Informationen suchen und finden*. Oldenbourg, 2007.

[Stu08] Sturm, M.: *Design und Implementierung des Onlinesystems Bewerbungsorganisator*. Otto-von-Guericke-Universität Magdeburg, Juli 2008.

[Stu09] Stuckenschmidt, H.: *Ontologien - Konzepte, Technologien und Anwendung*. Springer, 2009.

[TP06] Tassilo Pellegrini, A. B.: *Semantic Web - Wege zur vernetzten Wissensgellschaft*. Springer, 2006.

[VM06] Vincent Massol, T. M. O.: *Maven - Das Entwicklerheft*. O'Reilly, 2006.

[WGB08] Wolf Gideon Bleek, H. W.: *Agile Softwareentwicklung - Werte, Konzepte und Methoden*. dpunkt.verlag, 2008.

[Wik09] Wikipedia: *http://de.wikipedia.org/wiki/Semiotisches Dreieck*. letzter Besuch - 13.06.09, 2009.

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur mit erlaubten Hilfsmitteln angefertigt habe.

Magdeburg, den 18. August 2009

Vorname Name