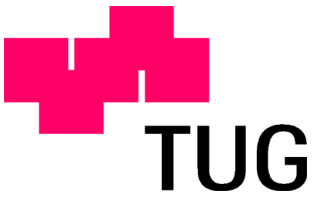
Masterarbeit

# Umsetzung einer Web 2.0 Applikation zur Diskussion von strukturierten Informationen.

Philipp Ast

---

Institut für Wissensmanagement
Technische Universität Graz
Vorstand: Univ.-Prof. Dr. rer.nat. Klaus Tochtermann




Begutachter: Univ.-Prof. Dr. rer.nat. Klaus Tochtermann
Betreuer: Dipl.-Ing. Dr. techn. Michael Granitzer

Graz, Februar 2010

# Abstract

Since more and more companies participate in business networks and therefore share information with other companies, a huge amount of the companies' key success factors is out of their actual operational sphere. Therefore, it is more important than ever to collect, aggregate and process information about the objectives and views of core partners and stakeholders.

The process of information collection can be performed in many different ways. Especially Web 2.0 offers possibilities to enrich internal, structured information. This enrichment is the main objective of this thesis and it is derived from a use case in which success factors and its connections in companies are considered. In the course of this thesis, it is investigated how popular Web 2.0 discussion platforms like wikis or blogs can be used to enrich existing structured information through external users.

As a solution, a Rich Internet Application to discuss structured information is proposed and developed. The application is basically a search engine which exploits user ratings as a form of discussion in contrast to traditional discussion platforms like blogs. The platform is evaluated in the context of a pilot project together with experts' interviews.

This thesis shows that with Rich Internet Applications, a high degree of usability is achieved. Also, the experiments showed that Rich Internet Applications can successfully be used to enrich structured information in form of success factors with external data.

For this thesis, the SUCCON Schachner & Partner KG made accessible selected conceptual and development activities that were accomplished during the project.

**Keywords:** Structured Information, Web 2.0, Rich Internet Applications

# Kurzfassung

Die zunehmende Vernetzung und Öffnung von Unternehmen nach extern bedingt, dass ein großer Teil erfolgsbestimmender Einflussfaktoren außerhalb des eigenen Wirkungsbereichs liegt. Deswegen ist es für erfolgreiches Management wichtiger denn je, Informationen zu den Sichtweisen und Motivationsfaktoren der zentralen, externen Stakeholder so zu erheben, dass sie mit den eigenen Sichtweisen zusammengeführt und weiterverarbeitet werden können.

Die Informationen können auf verschiedene Weise eingeholt werden. Gerade das Web 2.0 bietet hier Chancen, unternehmensinterne, strukturierte Informationen mit weiterführenden anzureichern. Diese Anreicherung ist zentraler Betrachtungspunkt der Arbeit und leitet sich aus einem speziellen Anwendungsfall ab, in welchem Erfolgsfaktoren und deren Verbindungen in Unternehmen betrachtet werden. Es wird untersucht, wie durch Diskussion über gängige Web 2.0 Plattformen wie Blogs oder Wikis etc. vorhandene, strukturierte Informationen durch Dritte angereichert werden können.

Als Lösungsansatz wird eine Plattform zur Diskussion strukturierter Informationen als Rich Internet Application konzipiert und entwickelt. Diese hat den Charakter einer Suchmaschine und im Gegensatz zu herkömmlichen Diskussionsplattformen, wie z.B. Blogs, erfolgt die Diskussion strukturiert mittels Bewertungen. Im Rahmen einer Pilotnutzung und Expertenbefragung wird diese evaluiert.

Die vorliegenge Masterarbeit zeigt, dass mit Rich Internet Applications ein hohes Maß an Usability erreicht werden kann, um strukturierte Informationen in Form von Erfolgsfaktoren erfolgreich mit weiterführenden Informationen durch Dritte anzureichern.

Der vorliegenden Masterarbeit liegt ein Auftrag der SUCCON Schachner & Partner KG an die TU Graz zugrunde. Einzelne der im Rahmen dieses Auftrages durchgeführten Konzeptions- und Entwicklungstätigkeiten wurden in Absprache mit der SUCCON zur Behandlung im Rahmen der vorliegenden Masterarbeit freigegeben.

**Stichwörter:** Strukturierte Informationen, Web 2.0, Rich Internet Applications

# Danksagung

An dieser Stelle möchte ich mich bei meinen Betreuern Prof. Tochtermann und Dr. Michael Granitzer bedanken, dass sie mir die Gelegenheit gegeben haben, diese Arbeit in der vorliegenden Art durchzuführen. Die Möglichkeit zu spontanen Diskussionen und hilfreiche Ratschläge waren ein wichtiger Faktor bei der Durchführung dieser Arbeit. Weiters bedanke ich mich bei allen Mitarbeitern des Know-Center Graz für deren Unterstützung und Hilfsbereitschaft. Großer Dank gilt auch Herrn Dr. Werner Schachner, der es mir ermöglichte, diese Masterarbeit im Umfeld der SUCCON umzusetzen. Er hat mir durch lange Diskussionen und nützliche Ratschläge maßgeblich weitergeholfen.

Danken möchte ich auch meinem Vater und meiner Mutter, die mich immer unterstützt haben, und ohne die mein Studium nicht auf diese Weise möglich gewesen wäre.

Besonderer Dank gilt meiner Freundin, die während meines gesamten Studiums viel Geduld mit mir bewiesen hat.

Graz, Februar 2010 Philipp Ast

Deutsche Fassung:
Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008
Genehmigung des Senates am 1.12.2008

## Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Graz, am ..... .....
(Unterschrift)

Englische Fassung:

## Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

..... .....
date signature

# Inhaltsverzeichnis

# Abbildungsverzeichnis

# Listings

# Kapitel 1

# Einleitung

## 1.1 Motivation

Das Management im 21. Jahrhundert sieht sich mit immer größer werdender Komplexität und Dynamik konfrontiert. Auf Management- und Führungsebene ist man in diesem Zusammenhang gefordert, die Funktionsweise und Erfolgsbasis des eigenen Geschäfts besser denn je zu kennen, um so das nötige Maß an zielgerichteter Entwicklungs- und Transformationsfähigkeit zu sichern. Eine zentrale Rolle spielt dabei die Kenntnis über die Erfolgsfaktoren des eigenen Geschäfts sowie deren wechselseitige Beziehungen.

Da sich Unternehmen in zunehmendem Maße vernetzen und dabei häufig nach extern öffnen, liegen auch die erfolgsbestimmenden Einflussfaktoren zu einem immer größer werdenden Teil außerhalb des eigenen, unmittelbaren Wirkungsbereichs. Deshalb sind künftig bei der Analyse der Funktionsweise des eigenen Geschäfts nicht nur die eigene Sichtweise, sondern auch die der zentralen Partner und Kunden zu berücksichtigen. Erfolgreiches Management wird sich vor allem darin begründen, Informationen zu den Sichtweisen, Einstellungen und Motivationsfaktoren der zentralen, externen Stakeholder so zu erheben, dass diese mit den Informationen zu den internen Sichtweisen, Einstellungen und Motivationsfaktoren zusammengeführt und gemeinsam weiterverarbeitet werden können (Stichworte: Ganzheitliche Geschäftsmodellanalyse, Querdenken und Innovieren).

Während die internen Informationen eigenen Denkmodellen entspringen und somit in bekannten und definierten Strukturen vorliegen, basieren die externen Informationen auf gänzlich anderen Denk- und Strukturmodellen (nämlich jener der jeweiligen Stakeholder). Und eben darin liegt eine der großen Herausforderungen im Management des 21. Jahrhunderts: Die Anreicherung strukturierter Daten aus internen Datenquellen mit Daten externer Quellen, ohne dabei die interne Struktur offen legen zu müssen.

Diese Motivation entspringt einem Auftrag der SUCCON Schachner & Partner KG an die TU Graz zur Durchführung einer Feasibility Studie zum Thema "Webbasierte Softwareunterstützung von Erfolgsdiagnosen". Einzelne der im Rahmen dieses Auftrages durchgeführten Konzeptions- und Entwicklungstätigkeiten (Praktische Umsetzung einzelner Teile zur Überprüfung der technischen Machbarkeit bzw. im Sinne eines Proof of Concept) wurden in Absprache mit der SUCCON zur Behandlung im Rahmen der vorliegenden Masterarbeit freigegeben. Dieser im Rahmen des Auftrages der SUCCON an die TU Graz umgesetzte Anwendungsfall wird im folgenden Kapitel näher erläutert.

### 1.1.1 Anwendungsfall

Die SUCCON Schachner & Partner KEG (kurz SUCCON) hat eine Methode zur Unternehmensberatung namens Erfolgsdiagnose®selbst entwickelt. Mithilfe der Erfolgsdiagnose ist es möglich, die Logik, die hinter dem Erfolg eines jeweiligen Betrachtungsgegenstandes steht, zu erheben und in Form eines Wirkungsdiagramms zu visualisieren. Ein Wirkungsdiagramm bildet die Erfolgsfaktoren sowie deren wechselseitige Wirkungsbeziehungen ab. Die SUCCON entwickelt zur Zeit eine webbasierte Software zur Unterstützung der Erfolgsdiagnose. Abbildung 1.1 gibt einen Überblick über das Vorhaben. Die Softwareanwendung soll eine Datenerhebung, -aufbereitung und -analyse für einzelne Diagnosefälle ermöglichen. Auch soll es breiten Usergruppen im Web ermöglicht werden, einzelne Daten aus verschiedensten Diagnosefällen einzusehen, um Ideen für eigene Vorhaben zu gewinnen. Im Gegenzug sollen diese externen Benutzer die Daten anreichern, um so andere Meinungen und Sichtweisen zu den Diagnosefällen zu erhalten. Basiernd auf bisherigen Erkenntnissen stellt diese Anreicherung von strukturierten Daten durch Dritte eine der größten Herausforderungen der SUCCON in der Entwicklung der Software dar. Aus dieser Ausgangssituation ergab sich ein spezieller Anwendungsfall, der in der vorliegenden Diplomarbeit behandelt und im Folgenden näher geschildert wird.

Betrachtet man Abbildung 1.1, so sind drei verschiedene Benutzer bzw. -gruppen erkennbar:

- Zum einen die **SUCCON**, welche die Plattform betreibt und die Kunden beratet. Im Rahmen der Beratung (Erfolgsdiagnose) wird der jeweilige Betrachtungsgegenstand erhoben und das Problem erarbeitet. Zusammen mit dem Kunden wird als Resultat ein Erfolgsprofil erstellt. Dieses wird zentral auf der Softwareplattform in einer Datenbank abgelegt.

- Die **Kunden** erhalten die Beratungsleistung von der SUCCON, können mit der Softwareanwendung bzw. Web-Plattform interagieren, auf die selbst erstellten Daten (Erfolgsprofile) zugreifen, diese bearbeiten und analysieren. Diese Funktionalitäten sind kostenpflichtig.

- **Interessierte Webnutzer** sind die dritte Benutzergruppe und die entscheidende für den Anwendungsfall (vgl. eingerahmten Bereich in Abbildung 1.1). Diese "externen" Nutzer können über eine Webschnittstelle auf einzelne Ausschnitte von Erfolgsprofilen zugreifen, haben jedoch aufgrund der Geheimhaltung der Daten keinen Vollzugriff auf die Erfolgsprofile. Diese sollen nach für sie interessante Informationen suchen, die Inhalte diskutieren sowie bewerten. Diese Webschnittstelle bietet demnach auch keine Analysemöglichkeiten wie der Kern der Anwendung, den bezahlende Kunden benutzen. Es wird eine Win-Win-Situation angestrebt, in der diese externen Webnutzer ihr Informationsbedürfnis befriedigen und als "Gegenleistung" die Informationen um weitere Facetten reicher machen sollen.

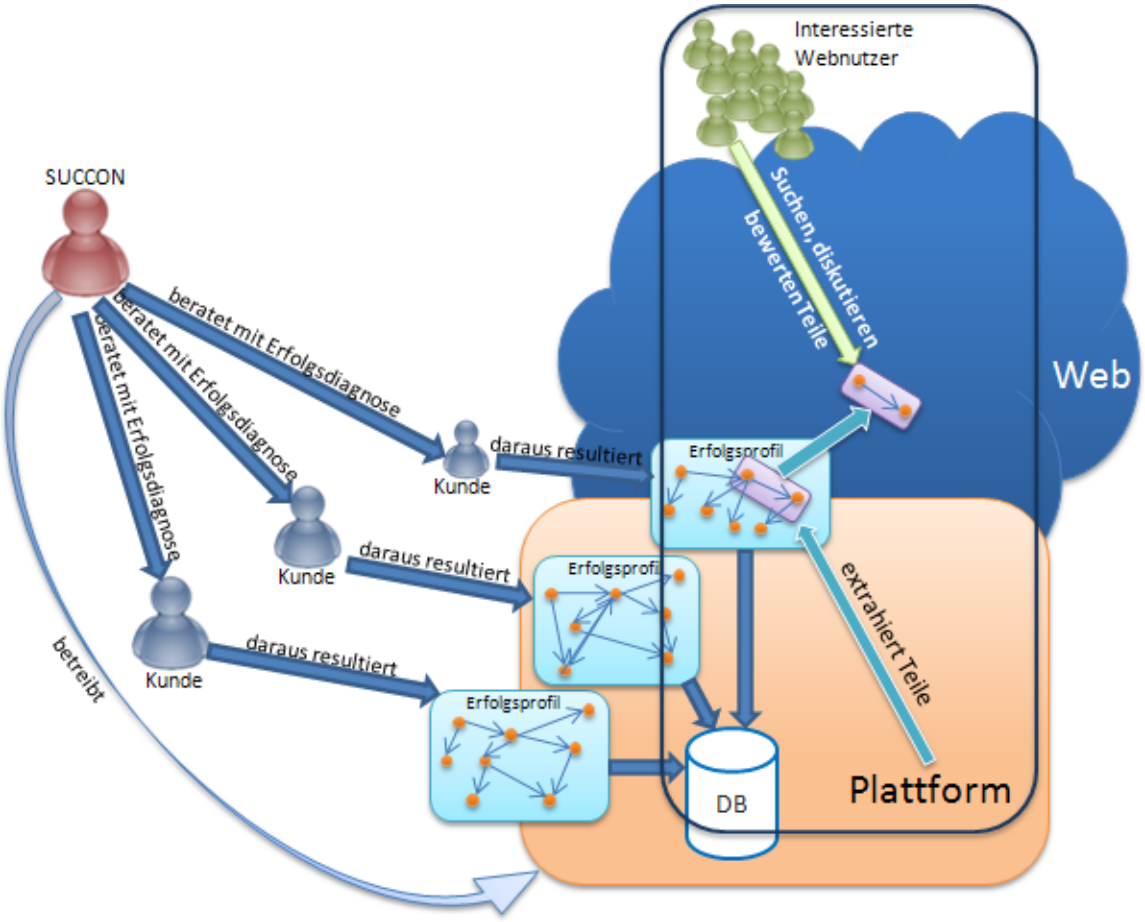



Abbildung 1.1: Überblick über die Softwareplattform und den Anwendungsfall

Im Rahmen der Erfolgsdiagnose werden Daten erstellt, aufbereitet und analysiert. Die drei wichtigsten Entitäten sind hierbei folgende: Erfolgsprofil, Erfolgsfaktoren und Erfolgsverbindungen. Daneben ist auch eine Stammdatenverwaltung (Benutzer,

Organisationen, etc.) notwendig. Nachfolgend werden die drei zentralen Informationsobjekte beschrieben:

- **Erfolgsprofil**: Ein Erfolgsprofil dient der Abbildung eines Betrachtungsgegenstandes, der im Rahmen einer Erfolgsdiagnose erhoben wird. Es stellt einen Beratungs- bzw. Diagnosefall dar und kann quasi als Container für dessen Erfolgsfaktoren und Erfolgsverbindungen gesehen werden. Ein Betrachtungsgegenstand kann ein gesamtes Unternehmen sein oder nur ein Teil daraus, ein Projekt, ein Geschäftsmodell oder ein akutes Problem, das erfasst werden soll. Ein Beispiel wäre "Erfolg im Netzwerkmanagement".

- **Erfolgsfaktor**: Ein Erfolgsprofil besteht aus mehreren Erfolgsfaktoren. Diese sind also jene Größen, die eine Wirkung auf den Erfolg eines Gegenstands haben. Die Erfolgsfaktoren sind die eigentlichen Informationsobjekte und beinhalten die tatsächlichen Informationen und Attribute wie z.B. einen Namen, eine fortlaufende Nummer oder einen Charakter (Stärke, Schwäche) und weitere. Um beim vorhin eingeführten Beispiel zu bleiben, könnten hier u.a. folgende Erfolgsfaktoren Teil des Erfolgsprofils sein: "Zufriedenheit der Mitarbeiter", "Mitarbeiterausbildung in Softskills", "Qualifikation der Mitarbeiter".

- **Erfolgsverbindung**: Mit der Erfolgsdiagnose werden nicht nur Erfolgsfaktoren definiert, sondern auch deren wechselseitige Wirkungsbeziehungen. Eine Erfolgsverbindung repräsentiert genau eine Wirkungsbeziehung zwischen zwei verschiedenen Erfolgsfaktoren. Diese Beziehungen sind gerichtet, d.h. die Richtung einer Beziehung ist entscheidend und die Unterscheidung in Quell- und Ziel-Erfolgsfaktor von Bedeutung. Wenn z.B. der Erfolgsfaktor A auf B wirkt, wirkt B *nicht* automatisch auf A. Dazu muss eine eigene Beziehung definiert werden. Drei Attribute sind für die Definition einer Erfolgsverbindung essentiell: Quelle, Ziel und Stärke. Eine Quelle ist ein Erfolgsfaktor, der auf einen Ziel-Erfolgsfaktor mit einer bestimmten Stärke wirkt. Diese Stärke kann einerseits positiv oder negativ sein und hat zusätzlich eine Intensität von stark oder schwach. Sie kann als ein Wert zwischen und 0 und 4 repräsentiert werden, wobei die Werte folgendes bedeuten: 0 ... stark negativ, 1 ... schwach negativ, 2 ... keine Wirkung[1], 3 ... schwach positiv, 4 ... stark positiv.
  Zum besseren Verständnis wird auch hier das vorige Beispiel aufgegriffen. Zum Beispiel können folgende Wirkungsbeziehungen definiert werden: Der Faktor "Zufriedenheit der Mitarbeiter" wird von den beiden anderen Erfolgsfaktoren beeinflusst, wirkt allerdings auf keinen der beiden. Somit ist er eine Zielfaktor. D.h. "Qualifikation der Mitarbeiter" wirkt genauso wie "Mitarbeiterausbildung in Softskills" positiv auf die "Zufriedenheit der Mitarbeiter", ersterer stark, zweiterer schwach positiv. Diese beiden sind somit Quellfaktoren. Die

[1] "keine Wirkung" bedeutet, dass keine Wirkungsbeziehung besteht. So eine Verbindung ist im eigentlichen Sinne keine Beziehung und kann daher gelöscht werden.

"Mitarbeiterausbildung in Softskills" wirkt zusätzlich positiv auf die "Qualifikation der Mitarbeiter".

### 1.1.2 Anwendungsszenarien für strukturierte Diskussion

Es werden nun zwei mögliche Szenarien für strukturierte Diskussion zum Anwendungsfall beschrieben. Das erste Szenario kombiniert die Eingabe von Erfolgsfaktoren zu einem bestimmten Thema mit einer Bewertung von bestehenden Erfolgsverbindungen. Die Benutzerschnittstelle teilt sich dabei in einen oberen und unteren Bereich und wird in Abbildung 1.2 angezeigt.

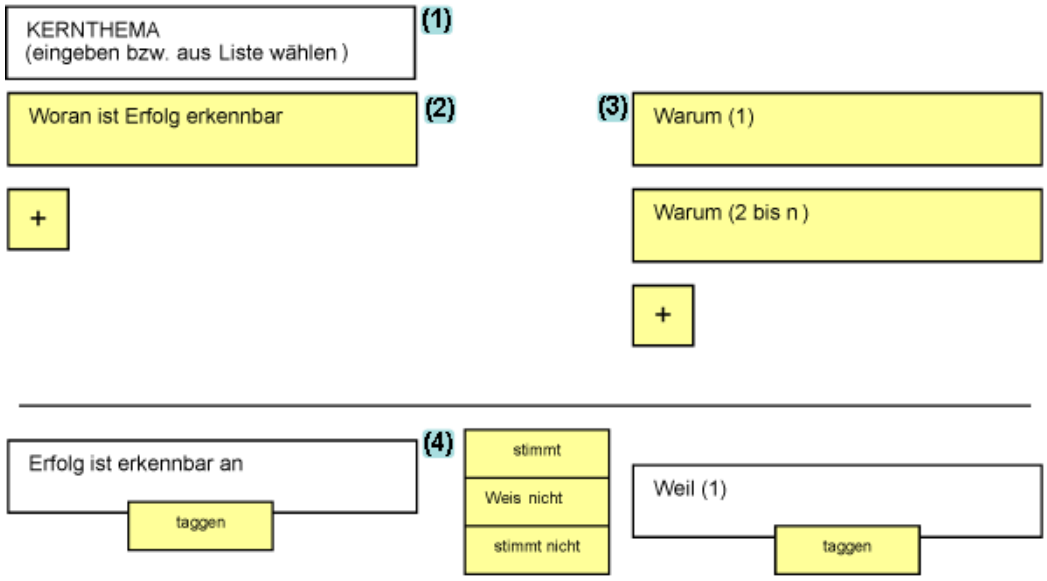



Abbildung 1.2: Mock-up: Suche, Eingabe und Bewertung des Einflusses von Erfolgsfaktoren

Dabei wählt der Benutzer ein Kernthema aus einer Liste aus oder sucht nach einem beliebigen Thema in einem Textfeld, zu dem er Informationen erhalten möchte (Abb. 1.2 (1) ). Nach erfolgter Themenauswahl muss er in je ein Textfeld eingeben, woran der Erfolg (in der gewählten Thematik) erkennbar ist (Abb. 1.2 (2) ). Diese eingegeben Punkte werden als Ziele (bzw. Zielfaktoren) bezeichnet. Für jeden Zielfaktor hat der User die Möglichkeit, eine maximal festgelegte Anzahl von Voraussetzungen (Quellfaktoren) einzugeben, welche die Zielfaktoren positiv beeinflussen (Abb. 1.2 (3) ). Für jede Voraussetzung erscheint im unteren Bereich eine bestehende Verbindung zweier Erfolgsfaktoren zum gewählten Thema (Abb. 1.2 (4) ). Der User hat nun die Möglichkeit, seine Meinung zur bestehenden Verbindung zu äußern, indem er sie für richtig oder falsch erklärt. Bei Unsicherheit kann er auch auf "weiß nicht" klicken. Zusätzlich kann der Benutzer auch Tags für die Erfolgsfaktoren der präsentierten Verbindungen vergeben. Dies dient einer weiteren Kategorisierung

und Beschreibung von Erfolgsfaktoren und kann z.B. für die zukünftige Suche oder Auswahl von Verbindungen nützlich sein.

In diesem Szenario werden nicht nur bestehende Informationen angereichert, sondern auch gänzlich neue hinzugefügt. Im oberen Teil werden Erfolgsfaktoren sowie Beziehungen erstellt. Jeder eingegebene Quellfaktor beeinflusst den dazugehörigen Zielfaktor, und für jedes dieser Tupel wird eine neue, gerichtete Verbindung angelegt. Im unteren Bereich wird bestehenden Verbindungen mehr oder weniger Gewicht verliehen, indem sie für richtig oder falsch erklärt werden. Diese Informationen werden auch strukturiert abgelegt und können so später in bestehende Erfolgsprofile einfließen. Z.B. kann sich der Ersteller des Erfolgsprofils die von externen Benutzern bewerteten Verbindungen anzeigen lassen. So ist ersichtlich, ob auch andere seiner Meinung sind, was die Verbindungen betrifft.

Probleme, die in dem Szenario auftreten, betreffen z.B. die Erstellung der Liste der möglichen Themen: Die vorhandene Struktur beinhaltet noch keine Kategorisierung der Erfolgsfaktoren bzw. -profile nach Themen, aber es kann z.B. der Name des Erfolgsprofils als Thema verwendet werden. Eine andere Option wäre die Verwendung von Tags (Schlagworten). Existierende Erfolsprofile werden dabei mit Tags versehen, und diese Tags werden zur Erstellung der Liste auswählbarer Themen herangezogen. So wird die Auswahl an möglichen Erfolgsfaktoren für die Darstellung der Verbindungen im unteren Bereich vergrößert, da ein und dasselbe Tag auf verschiedene Erfolgsprofile zutreffen kann. Der Missbrauch des Systems, nur um relevante Verbindungen präsentiert zu bekommen, ist ein erheblich größeres Problem. Denn die Behandlung der Eingabe neuer Erfolgsfaktoren (Quell- und Zielfaktoren) ist sehr diffizil. Das System müsste dabei (z.B. mit Techniken des Text Mining) unterscheiden, ob es sich um eine Falscheingabe handelt oder um wirklich relevante Begriffe. Falscheingaben sind zum einen Begriffe, die für das System bzw. im Kontext des Themas irrelevant sind (z.B. Hund, Katze) und zum anderen Begriffe ohne echte semantische Bedeutung (z.B. “bla bla”, “rktwtrm”). Zudem müssten auch Rechtschreibfehler bei relevanten Begriffen erkannt sowie Anglizismen berücksichtigt werden. Hinzu kommt, dass durch diesen Ablauf zwar viele Informationen auf einen Schlag generiert und bestehende Informationen angereichert werden, der Weg allerdings lang und uU. etwas kompliziert ist, bis ein Benutzer neue Informationen und somit einen Mehrwert erhält. Das heißt, die Usability der Anwendung leidet darunter.

Das zweite Szenario zur Anreicherung strukturierter Informationen ist wesentlich einfacher gehalten und hat den Charakter einer Suchmaschine. Diese wird in Abbildung 1.3 gezeigt. Hier können im Gegensatz zum vorigen Szenario keine neuen Erfolgsfaktoren eingegeben werden, der Benutzer bewertet nur bestehende Verbindungen mit einer mehrstufigen Skala. Die Auswahl der zu bewertenden Verbindungen trifft der Benutzer selbst, indem er einen Suchbegriff in ein Textfeld eingibt

und die Suchanfrage absendet (Abb. 1.3 (1) ). Als Suchresultat wird eine Liste bestehender Verbindungen angezeigt, die der Nutzer selbst bewerten kann (Abb. 1.3 (2) ). Für die Bewertung werden zwei Bewertungs-Balken mit einer diskreten Skala verwendet: ein Balken für jede Richtung, wobei ein Pfeil angibt, welcher Erfolgsfaktor auf welchen wirkt. Die Skala ist fünfstufig, wobei die Werte die möglichen Wirkungsbeziehungen zwischen zwei Erfolgsfaktoren repräsentieren: Eine Wirkung ist entweder positiv oder negativ und hat eine starke oder schwache Intensität. Z.B. wird "stark positiv" als "++" und "schwach negativ" als "-" symbolisiert (vgl. Abschnitt 1.1.1). Wenn laut Benutzer keine unmittelbare Wirkungsbeziehung besteht, so muss die Mitte (Wert "0") gewählt werden.

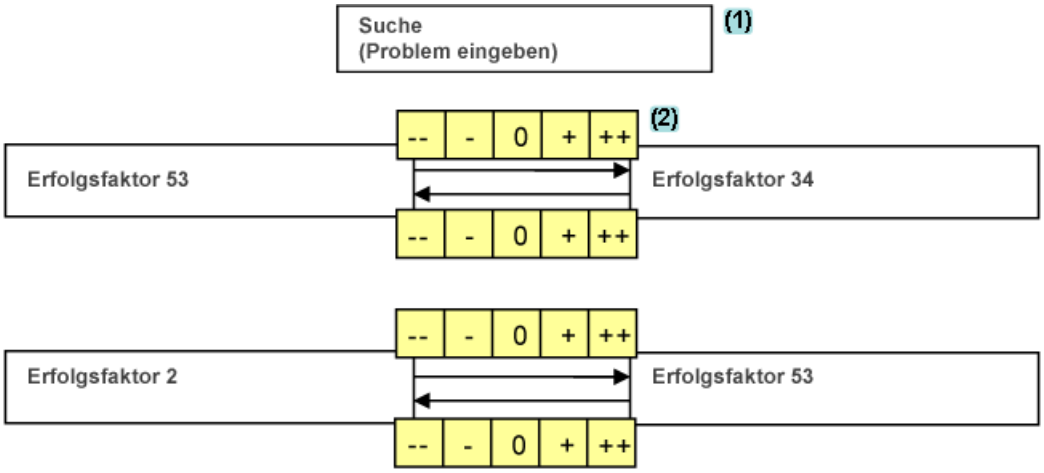



Abbildung 1.3: Mock-up: Bewertung der Beziehung von Erfolgsfaktoren

Um einen User zu motivieren, auch selbst Bewertungen abzugeben und nicht nur zu suchen, kann man verschiedene Be- bzw. Einschränkungen einführen. So könnte die Anzahl der freien Suchen für anonyme User begrenzt sein. Für eine weitere Nutzung danach muss sich der Benutzer registrieren bzw. anmelden. So könnten Such- und Bewertungshistorien erfasst werden, die in die Erstellung der Suchresultate einfließen, um relevantere Ergebnisse zu liefern. Um User zu Bewertungen zu bewegen, kann folgender Mechanismus verwendet werden: Nach einer bestimmten Anzahl an freien Suchen muss der Benutzer ein Verbindungspaar bewerten, um weiter suchen zu können. Danach wird dieser Zähler zurückgesetzt und er kann wieder eine bestimmte Anzahl an Suchen absetzen. Dieser Zähler könnte auch dynamisch anpassbar sein, d.h.: Je mehr Suchen er insgesamt abgesetzt hat, umso öfter (kürzeres Intervall zwischen Suche und Bewertung) oder umso mehr (gleichbleibendes Intervall zwischen Suche und Bewertung, es müssen allerdings mehr Bewertungen abgegeben werden) muss er bewerten. So wird sichergestellt, dass auch Informationen angereichert werden. Zusätzlich könnten Premium-Benutzer-Accounts angeboten werden, die gegen eine geringe, monatliche Gebühr zu haben sind, allerdings keine Bewertungen vor-

aussetzen und so eine "ununterbrochene Suche" ermöglichen.

Damit das System nicht ausgenutzt wird, ist es ratsam, pro Suchanfrage nur eine begrenzte Anzahl an Resultaten anzuzeigen. Dies hat nebenbei den positiven Effekt, dass der Benutzer nicht durch die etwas komplexe bzw. ungewohnte Darstellung der Resultate überfordert wird, und er die präsentierten Informationen entsprechend rezipieren kann. Da die Erfolgsfaktoren isoliert und ohne Kontext zum dazugehörigen Erfolgsprofil angezeigt werden, wird auch keine Strukturinformation preisgegeben, die Daten im Gesamten bleiben geheim. Natürlich dürfen diese Bewertungen die bestehenden Erfolgsfaktoren und -verbindungen nicht verändern sondern erweitern.

## 1.2 Zielsetzung und Fragestellung

Die vorliegende Masterarbeit soll im Rahmen des Anwendungsfalles der Fragestellung nachgehen, wie die Diskussion von strukturierten Informationen im Web 2.0 bzw. durch Web 2.0 Technologien bewerkstelligt werden kann. Das dahinterliegende Ziel ist die Anreicherung strukturierter Informationen mit weiterführenden durch Dritte unter der Prämisse, dass Benutzer das System bzw. den Kontext, in dem die Daten stehen, nicht näher kennen. Im konkreten Fall wird die strukturierte Diskussion anhand von Erfolgsfaktoren und -verbindungen behandelt.

Zur umfassenden Behandlung der Fragestellung muss im Vorfeld der theoretische Rahmen aufgespannt werden. Dabei werden das Web 2.0 im Detail und in weiterer Folge Diskussionsplattformen im Web 2.0 beschrieben. Die Arbeit konzentriert sich im Speziellen auf die folgenden Fragestellungen:

1. Welche Plattformen zur Diskussion von strukturierten Informationen existieren im Web 2.0 bereits?
2. Lassen sich diese zur strukturierten Diskussion bestehender Informationen aus dem Anwendungsfall nutzen?
3. Inwieweit können Web 2.0 Technologien dazu genutzt werden, die Anreicherung strukturierter Informationen aus dem beschriebenen Anwendungsfall mit Daten von Dritten unter der Restriktion der Geheimhaltung zu ermöglichen?

Der praktische Teil der Arbeit behandelt die Umsetzung einer Plattform zur Diskussion strukturierter Informationen im oben beschriebenen Anwendungsfall unter Verwendung von Web 2.0 Technologien.

## 1.3 Gliederung

Diese Arbeit gliedert sich in einen theoretischen Teil und die Beschreibung eines entwickelten Konzepts inklusive dessen Umsetzung als Web 2.0 Applikation, durch wel-

che strukturierte Informationen diskutiert und damit angereichert werden können.

Der theoretische Rahmen der Arbeit ist Gegenstand von Kapitel 2. Nach der Defintion des Begriffs Web 2.0 und der Beschreibung, was unter strukturierten Informationen verstanden wird, werden in Abschnitt 2.2 Plattformen zur Diskussion im Web 2.0 eingehend beschrieben. Unter anderem wird dabei für jede Plattform das Potenzial zur Diskussion der strukturierten Informationen aus dem Anwendungsfall erörtert.

In Kapitel 3 wird vom theoretischen in den praktischen Teil übergeleitet. Dabei werden eingangs Rich Internet Applications im Allgemeinen als gewählte Umsetzungsplattform beschrieben. Es folgt die Beschreibung des Konzepts, wie die strukturierten Informationen durch Dritte diskutiert und angereichert werden sollen.

Kapitel 4 umfasst die Implementierung des entwickelten Konzepts, welche eine detaillierte Beschreibung der technischen Umsetzung als Rich Internet Application beinhaltet. Das System wurde im Rahmen einer Pilotnutzung evaluiert, was in Kapitel 5 beschrieben wird. Den Abschluss der Arbeit bildet Kapitel 6 mit der Zusammenfassung und einem Ausblick für zukünftige Entwicklungen.

# Kapitel 2

# Diskussion von strukturierten Informationen im Web 2.0

In diesem Kapitel wird der theoretische Hintergrund der Arbeit behandelt. Eingangs werden zentrale Begriffe wie Web 2.0 und strukturierte Informationen definiert. Danach werden Plattformen zur Diskussion von strukturierten Informationen im Web 2.0 vorgestellt.

## 2.1 Definitionen und Begriffsabgrenzungen

### 2.1.1 Web 2.0

Das Schlagwort Web 2.0 bezeichnet das World Wide Web (kurz Web oder WWW) einer neuen Generation. Im folgenden wird kurz der geschichtliche Hintergrund in der Entwicklung hin zum Web 2.0 geschildert, danach werden die Prinzipien und Grundpfeiler von Web 2.0 erörtert.

#### 2.1.1.1 Vom Web 1.0 zum Web 2.0

[Lange, 2007] postuliert *"Der Webnutzer 2.0 surft nicht mehr nur durch das Web, sondern verändert und bereichert es."*. Das ist ein zentraler Unterschied in der Evolution des Webs: Im Web der ersten Generation ging es vor allem um das Finden von Information und das Navigieren durch das WWW. Das Gros an Inhalten wurde von Unternehmen produziert und im Web publiziert. Das Erstellen von Webseiten und -applikationen war teuer und schwierig zugleich. Nur wenige versierte Benutzer hatten die Möglichkeiten und Fähigkeiten, Inhalte zu veröffentlichen. Abgesehen davon war auch die Anzahl der Internetbenutzer in den Neunziger Jahren bedeutend geringer als heute (vgl. [Bettel, 2008], [Alby, 2007]).

Das Platzen der Dotcom-Blase im Herbst 2001 markiert einen Wendepunkt in der Entwicklung des Webs. Die Dotcom-Blase war ein weltweites Phänomen, das vor

allem so genannte Dotcom-Unternehmen[1] betraf. Internet-Unternehmen drangen an die Börse, oftmals ohne funktionierendes Geschäftsmodell, jedoch mit hohen Gewinnerwartungen. Viele Internet-Aktien waren maßlos überzeichnet, und als die Gewinnerwartungen der Unternehmen nicht erfüllt und erste Insolvenzen angemeldet wurden, sanken die Aktienkurse ab März 2000 drastisch. In weiterer Folge brach der Aktienmarkt ein und wenige erfolgreiche Dotcom-Unternehmen überlebten [Glebe, 2008].

Doch das Web und auch dessen Umfeld änderten sich: Datenübertragungsraten wurden schneller, Breitbandanschlüsse mit einer Geschwindigkeit von mehreren MegaBit/s sind mittlerweile der Standard. Anschlüsse wurden aber auch billiger, und somit war die Eintrittsbarriere ins WWW geringer. Dies hatte zur Folge, dass die Anzahl der Internetsurfer stark anstieg [Alby, 2007]. Dale Dougherty und Craig Cline erwähnten im Rahmen eines Brainstormings zwischen O'Reilly und MediaLive International im Jahre 2004 erstmals den Begriff Web 2.0. Sie erkannten, dass das Web lebendiger und wichtiger als je zuvor war: Die Geschäftsmodelle der im Web erfolgreich tätigen Unternehmen änderten sich [O'Reilly, 2005]. Neue, innovative Anwendungen wurden erstellt, wie z.B. 2003 die Social Networking Plattform MySpace[2]. User können damit auf einfachste Weise ein Benutzerprofil erstellen, Netzwerke zu Freunden aufbauen und Blogs, Bilder sowie Videos veröffentlichen. 2004 ging die Foto-Sharing-Webseite Flickr online. Mit einem einfach zu bedienenden User Interface können die Benutzer Fotos uploaden, bearbeiten, mit Schlagworten versehen und anderen Usern zugänglich machen. Diesen beiden Beispielen folgten viele weitere innovative Applikationen, die im Web zugänglich sind: Die E-Mailwebanwendung GMail[3], die Social News Anwendung Digg[4] oder die Videoplattform YouTube[5], um nur einige weitere erfolgreiche Web 2.0 Anwendungen zu nennen [Informationweek, 2006].

Was allen diesen Webanwendungen gemein ist, ist die Tatsache, dass die Inhalte von den Benutzern und nicht etwa von professionellen Autoren erstellt werden. [Vickery and Wunsch-Vincent, 2007] spricht in diesem Zusammenhang auch von "user-generated content" (oder auch "user-created content"). Es hat also ein Paradigmenwechsel in der Benutzung des Webs stattgefunden: Statt nur Konsument zu sein, ist der Benutzer auch Produzent von Inhalten. Nach [Toffler, 1980] nennt man diese auch Prosumer. Jeder Benutzer kann heute auf einfache Weise an der Gestaltung des Webs partizipieren, deswegen auch der Name partizipatives Web oder Mitmach-Web (vgl. [Murugesan, 2007], [Peters and Stock, 2007] ).

---

[1] Unternehmen, die Dienstleistungen im Zusammenhang mit dem Internet anbieten

[2] `http://www.myspace.com/`

[3] `http://mail.google.com/`

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

[5] `http://www.youtube.com`

Tim O'Reilly war es letztlich, der den Begriff Web 2.0 der Internetgemeinde im Jahr 2005 vorstellte. Er verfasste einen Essay [O'Reilly, 2005], in dem die Gemeinsamkeiten erfolgreicher Unternehmungen im Web wie Google, Amazon oder Ebay uvw. in sieben Prinzipien und Designmuster zusammengefasst sind. O'Reilly spannt in seinem Essay den Rahmen von Web 2.0 auf und geht im Detail auf die damit verbundenen Thematiken ein, eine kurze, prägnante Defintion bleibt der Autor allerdings schuldig. [Musser and O'Reilly, 2006] finden eine klare Definition von Web 2.0:

> *"Web 2.0 is a set of economic, social, and technology trends that collectively form the basis for the next generation of the Internet - a more mature, distinctive medium characterized by user participation, openness, and network effects."*

Es handelt sich also bei Web 2.0 um das Internet einer neuen Generation, das aus einer Menge von ökonomischen, sozialen und technologischen Trends entstanden ist. Dieses weiterentwickelte Web zeichnet sich vor allem durch Benutzerpartizipation, Offenheit und Netzwerkeffekte aus. Im folgenden Abschnitt werden die Facetten des Web 2.0 eingehender diskutiert, um den Erfolg von Web 2.0 in seiner vollen Bandbreite zu erfassen.

#### 2.1.1.2 Der Erfolg von Web 2.0

Nach [O'Reilly, 2005] folgen erfolgreiche Web 2.0 Unternehmen bestimmten Prinzipien und bewährten Methoden, sog. Best Practices. Diese Prinzipien und Best Practices wurden in [Musser and O'Reilly, 2006] aufgegriffen und überarbeitet. Im folgenden werden diese näher betrachtet und somit erklärt, was den Erfolg von Web 2.0 bzw. Web 2.0 Anwendungen ausmacht.

**Kollektive Intelligenz**
[Surowiecki, 2005] ist der Ansicht, dass Gruppen unter richtigen Umständen intelligenter sind als die Gescheitesten in ihrer Mitte und schreibt über die kollektive Intelligenz:

> *"Für das kluge Verhalten solcher Gruppen bedarf es eben nicht dominanter, außergewöhnlich gescheiter Mitglieder. Mit anderen Worten: Auch wenn die allermeisten Angehörigen einer Gruppe weder sonderlich informiert noch zu rationalem Denken imstande sind, vermögen sie als Kollektiv gleichwohl vernünftige Entscheide zu treffen."*

Angelehnt an Surowiecki formulieren [Musser and O'Reilly, 2006] das Nutzen dieser kollektiven Intelligenz als das wichtigste Prinzip im Web 2.0. Erfolgreiche Web 2.0 Applikationen schaffen ihrer Ansicht nach eine Architektur der Partizipation, um die kollektive Intelligenz zu nutzen. Beteiligen sich Benutzer am Erstellen von Inhalten, so werden die Benutzer und Gruppen einer Webapplikation zum Motor

der Innovation für bessere Produkte und von schnellerem Wachstum. Dies schafft wiederum Kundenzufriedenheit und -vertrauen in eine Applikation.

Die Beteiligung der Benutzer zu gewinnen ist aufgrund von zwei Faktoren bedeutend: Erstens schaffen Benutzer einen Mehrwert, zweitens wird dieser Mehrwert durch Netzwerkeffekte vergrößert. Der Mehrwert wird durch den Benutzer direkt geschaffen, indem er z.B. Inhalte erstellt, Kommentare abgibt oder Videos oder Fotos hochlädt. Indirekt wird Mehrwert geschaffen, indem z.B. das Surfverhalten auf einer Webapplikation protokolliert wird, um z.B. die am meisten verwendeten Funktionen zu eruieren.

Netzwerkeffekte treten auf, wenn ein Produkt oder ein Service wertvoller wird, je mehr Personen es verwenden. Nach dem Reed'schen Gesetz steigt die Nützlichkeit großer Netzwerke exponentiell mit ihrer Größe[Reed, 1999]. Dies gilt auch für soziale Netzwerke, wie sie im Web 2.0 vorkommen, z.B. auf der Social Networking Seite Facebook[6] oder Myspace. Am Beispiel der Online Enzyklopädie Wikipedia[7] soll dies näher erläutert werden: Wikipedia ist die mittlerweile umfassendste Enzyklopädie weltweit. Dies ist möglich geworden, da jeder Benutzer jeden Artikel lesen sowie bearbeiten und auch neue Artikel verfassen kann. Abgesehen davon werden die Artikel meist von einer Gruppe von Personen gepflegt, was zur Folge hat, dass nicht nur eine Ansicht zu einem Thema vertreten ist, sondern die der Masse. Damit werden Einträge umfangreicher und falsche Informationen i.d.R. ausgebessert. Dass die Qualität der Artikel hoch ist, wurde in einem Vergleich mit der kommerziellen Enzklopädie Britannica bewiesen[Terdiman, 2005]. Je mehr Artikel in der Wikipedia verfügbar sind, desto eher werden auch gesuchte Informationen gefunden. Das hat zur Folge, dass mehr Internet User auf Wikipedia zugreifen und es als Nachschlagewerk verwenden. Diese wiederum beteiligen sich unter Umständen wieder daran, Artikel zu verfassen oder zu bearbeiten. Wikipedia ist ein gelungens Beispiel, das exakt dem Web 2.0 Leitsatz von [O'Reilly, 2006] entspricht: "Build applications that harness network effects to get better the more people use them".

Ein weiteres Beispiel, das Netzwerkeffekte erfolgreich nutzt, ist Amazon[8], das größte Online-Versandhaus der Welt. Amazon hat es als erster Onlinehändler verstanden, Benutzer aktiv partizipieren zu lassen und somit zum Branchenprimus zu werden. Anstatt professionelle Rezensoren und Autoren einzusetzen, verfassen Benutzer Rezensionen selbst und können Bewertungen zu Produkten abgeben. Die Rezensionen und Bewertungen werden direkt auf der Produktseite angezeigt. Daneben können Benutzer u.a. Bilder zu Produkten hochladen, Produktwikis bearbeiten oder auch Benutzerhandbücher verfassen. Somit wird der Produktkatalog mit weiterführender Information angereichert, es entsteht ein eindeutiger Mehrwert. Indirekt stiften

[6] `http://www.facebook.com/`
[7] `http://de.wikipedia.org`
[8] `http://www.amazon.com/`

Benutzer einen Mehrwert, indem ihr Kaufverhalten aufgezeichnet und für ein Empfehlungssystem eingesetzt wird. Dies sind nur einige der Funktionen, die Amazon einsetzt, welche es zum erfolgreichsten Online-Versandhaus werden ließen.

**Daten als "Intel inside"**
Der Name dieses Prinzips Daten als "Intel inside" rührt von einem Vergleich mit Intels Werbekampagne der letzten 15 Jahre. Mit dem Slogan postuliert Intel, dass in fast jedem PC System zwar verschiedene proprietäre Hardware Komponenten stecken (Festplatten, Grafikkarten, Motherboards u.d.g.), jedoch die CPU nur von einem Hersteller stammt, nämlich Intel. Durch diese Strategie errang Intel eine Vormachtstellung auf dem Mikroprozessor-Markt[Musser and O'Reilly, 2006].

Ähnlich verhält es sich in der datenzentrierten Internet Ära mit den Daten, denn erfolgreiche Webapplikationen glänzen nicht nur durch Funktionen: Google verfügt über den größten Suchindex, Amazon hat eine riesige Produktdatenbank und Flickr verfügt über eine umfangreiche Bilddatenbank. Ohne diese Daten würden kaum so viele User diese Anwendungen verwenden. Durch den vollzogenen Marktwandel von reinen Desktop Applikationen hin zu Webapplikationen und Online Services gewannen die Daten und die Kontrolle darüber an Wert und Bedeutung und stellen somit einen entscheidenden Wettbewerbsvorteil gegenüber Konkurrenten dar[O'Reilly, 2005].

Tele Atlas und NAVTEQ sind die zwei größten Anbieter für Geodaten und digitales Kartenmaterial. So versorgt Tele Atlas den Online Kartendienst Google Maps[9], während NAVTEQ die Konkurrenten Yahoo! Maps, Bing Maps oder MapQuest[10] mit Kartenmaterial beliefert. Die beiden Anbieter nehmen eine Schlüsselrolle für Webmapping Dienste ein, da sie Daten besitzen, die einerseits teuer in der Erstellung und andererseits schwer zu reproduzieren sind. Im Gegensatz dazu kann durch Netzwerkeffekte eine große Datenbank bzw. ein wertvoller Datenpool auf billigere Weise generiert werden, wie dies eBay[11] (Produkt- und Verkäuferdatenbank), YouTube (Videos) oder delicious[12] (Bookmarks, Lesezeichen) erfolgreich praktiziert haben.

Unternehmen verfolgen auch andere Strategien, um mittels Daten einen strategischen Vorteil zu erzielen[Musser and O'Reilly, 2006]. Kontrollstrategien werden ermöglicht durch proprietäre Dateiformate oder Datenzugriffsmechanismen über spezielle Verzeichnisse, wie z.B. Gracenote[13]. Gracenote verfügt über eine Daten-

[9]Webseite: `http://maps.google.com/`; Google war bis September 2008 Kunde von NAVTEQ, wechselte allerdings zum Konkurrenten Tele Atlas[Tele-Atlas, 2008], nachdem der Mobiltelefon-Hersteller Nokia NAVTEQ übernahm[Nokia, 2007]

[10]`http://maps.yahoo.com/`, `http://www.bing.com/maps/`, `http://www.mapquest.com/`

[11]`http://www.ebay.com/`

[12]`http://delicious.com/`

[13]Webseite: `http://www.gracenote.com`; vormals Compact Disc Database (CDDB)

bank, in der Informationen über sämtliche am Markt befindlichen Audio-CDs gespeichert sind und über das Internet abgerufen werden können.
Limelight Networks [14] verfolgt Daten-Infratstruktur-Strategien und verfügt über sog. Content Distribution Networks zur Bereitstellung von großen Mediendateien wie Audio- oder Videodateien sowie anderen Inhalten, vor allem für Großkunden[15] wie Microsoft, DreamWorks SKG oder Disney.
EveryBlock[16] verfolgt eine sogenannte Zugriffsstrategie: Über die Webseite wird der Zugriff auf tagesaktuelle Verbrechensstatistiken oder Restaurantinspektionen von 15 verschiedenen Großstädten der USA[17] ermöglicht. Somit sind User über das Geschehen in der Nachbarschaft oder in restlichen Stadtteilen informiert. Das Neue daran ist, dass diese Daten bisher nicht verfügbar waren, und EveryBlock diese Daten sammelt, aggregiert und den Benutzern zur Verfügung stellt.

**Das Web als Plattform**
In der PC Ära schaffte es Microsoft immer wieder durch die weite Verbreitung und proprietäre Schnittstellen (Application Programming Interfaces, APIs) der Windows Plattform Platzhirsche zu verdrängen: So wurde das beliebte Textverarbeitungsprogramm WordPerfect durch Microsoft Word ebenso abgelöst, wie der Browser Netscape Navigator durch den Microsoft Internet Explorer. Doch durch offene Standards und Protokolle wie TCP/IP, HTTP und XML wurde das Web selbst zu einer Plattform für Anwendungen. Zudem erhalten Webseiten ihrerseits selbst Plattformcharakter durch das Bereitstellen von offenen APIs. Durch die APIs kann auf die Daten und Services von Anbietern von Webangeboten zugegriffen werden, ohne mit der Webseite oder -applikation selbst zu interagieren[O'Reilly, 2005].

Bieten Hersteller APIs zu ihren Services an, so fördert dies Innovation durch Dritte, schafft Vertrauen und hilft beim Aufbau von Communities. Durch Statistiken erfährt man zusätzlich, wie Services wirklich genutzt werden[Musser and O'Reilly, 2006]. Der Verzeichnisdienst ProgrammableWeb verfügt über eine umfangreiche Liste mit ca. 1400 verschiedenen APIs[18], u.a. eBay, Digg, delicious, Flickr, Google. Dass APIs auch tatsächlich einen Mehrwert bringen und benutzt werden, zeigt ein Blick auf die Benutzungsstatistik von eBay's API: So wurden bis Dezember 2005 47% aller Artikel zu den eBay Versteigerungen über die Web Service API hinzugefügt. Im vierten Quartal des Jahres 2005 wurden mehr als 8 Milliarden Anfragen über die Web Service API gestellt[Research, 2006].

Durch das Einbinden und Kombinieren von APIs in bestehende Webseiten entstand eine neue Art von Webapplikationen: Mashups. Ein Mashup nutzt die offenen

[14] `http://www.limelightnetworks.com`
[15] Kundenliste: `http://www.limelightnetworks.com/customers/`
[16] `http://www.everyblock.com`
[17] Stand: Juli 2009, siehe: `http://www.everyblock.com/about/faq/`
[18] `http://www.programmableweb.com/apis/directory`, Stand: 30. Juli 2009

APIs und bereichert somit die eigene Webapplikation mit weiteren Daten. So lassen sich auch Anwendungen erstellen, die nur aus APIs von vorhandenen Applikationen bestehen[Alby, 2007]. Im Jahr 2005 ging HousingMaps.com[19] als erstes Mashup online. Es verwendet Immobilien Daten von Craigslist.com[20] und kombiniert diese mit der Google Maps API[21], um die Lage der Immobilien auf einer Karte zu visualisieren. Die Applikation BBC News Map[22] verwendet ebenso die Google Maps API: Auf einer Karte wird angezeigt, an welchem Ort ein Ereignis (bzw. ein News-Eintrag) passiert ist. Der Vorteil liegt auf der Hand: Mit einem Blick kann man feststellen, ob es Neuigkeiten in einem für den User interessanten Gebiet gibt.
Ein Verzeichnis mit über 4400 Mashups ist unter ProgrammableWeb.com[23] verfügbar: Den Löwenanteil von verwendeten APIs in Mashups machen dabei Mapping APIs (36%, z.B. Google Maps, Live Maps) aus, gefolgt von Photo APIs (10%, z.B. Flickr, Phootbucket) und Shopping APIs (9%, z.B. Amazon, eBay). Erst danach folgen Such APIs (8%, z.B. Google, Yahoo, Bing) sowie Video APIs (8%, z.B. YouTube, Netflix).

**Dynamische Benutzerschnittstellen und bessere Benutzerführung**
Das Web 2.0 manifestiert sich in Webapplikationen, die reaktionsschnell und in der Regel einfach zu bedienen sind. Reaktionsschnell bedeutet in diesem Kontext kurze Wartezeiten für den Benutzer, nachdem Aktionen ausgeführt wurden. Webapplikationen werden mit einer Technik namens "Asynchronous JavaScript and XML" (kurz AJAX) oder auch mit der Adobe Flash Plattform[24] realisiert. Im Gegensatz zum traditionellen Modell einer Webanwendung wird bei AJAX Anwendungen nicht die gesamte Seite nach dem herkömmlichen Request-Response-Modell neu geladen, sondern nur Teile der bestehenden Seite. Dadurch wird das Anwendungsverhalten verbessert sowie ein flüssigeres Arbeiten ermöglicht[Garrett, 2005].

Sogenannte "Rich User Experiences" sind nicht mehr Desktop Anwendungen vorbehalten, auch mittels Webapplikationen können hoch interaktive Benutzerschnittstellen und Funktionen wie z.B. Drag und Drop einfach realisiert werden. Im Gegensatz zu Desktop Applikationen müssen Anwendungen im Web nicht installiert werden. Darüberhinaus kann mittels eines Browsers überall darauf zugegriffen werden. Für Webapplikationen, welche auf der Adobe Flash Plattform laufen, muss zwar der Adobe Flash Player[25] installiert werden. Doch dies muss nur einmal gemacht werden, alle auf dieser Plattform basierenden Anwendungen sind dann im Browser lauffähig. Eine Webapplikation ermöglicht zudem auch kollaboratives Arbeiten. Zusammenfassend kombinieren Web 2.0 Applikationen die Vorteile von Software aus dem Offline-

[19] `http://www.housingmaps.com/`
[20] Der Zugriff auf Daten von Craigslist.com funktioniert über die RentRent API: `http://www.rentrent.org/RENT/API/index.html`
[21] `http://code.google.com/apis/maps/`
[22] `http://dev.benedictoneill.com/bbc/`
[23] `http://www.programmableweb.com/mashups/directory/`, Stand: 13. November 2009
[24] `http://www.adobe.com/flashplatform/`
[25] `http://www.adobe.com/products/flashplayer`

und Online-Modell[Musser and O'Reilly, 2006].

**Software über Gerätegrenzen hinweg**
Heutzutage sind nicht mehr der PC oder Laptop die einzigen Geräte, mit dem Benutzer auf das Internet zugreifen. Mittlerweile konsumiert man Angebote aus dem Internet auch via tragbare Medien Player (z.B. Apple iPod), Spielekonsolen (z.B. Microsoft Xbox) oder auch Mobiltelefone. Aus diesem Grund verschafft man sich einen Wettbewerbsvorteil, wenn man Software anbietet, die so entworfen ist, dass auf dessen Services und Daten Benutzer nicht nur mit Computern zugreifen können[Musser and O'Reilly, 2006].

Apple's iTunes[26] ist ein Paradebeispiel für geräteübergreifende Software. Es ist zwar keine Web 2.0 Applikation, jedoch nutzt sie das Web als Plattform. Zum einen dient die Anwendung als Mediathek (Medienbibliothek), die Musik und Videos verwaltet. Zum anderen wird mit iTunes Musik auf den iPod oder das iPhone übertragen und dient als Kontrollstation für Apples tragbare Medienplayer. ID3 Tags für Musikdateien und Metadaten für Musikalben wie Bilder der Covers können über das Web heruntergeladen werden, wobei die Applikation auf den Apple iTunes Store zugreift. Über diesen können Musik, Podcasts, Videos und weitere Mediendateien gekauft und heruntergeladen werden. Wird ein Musikstück über den iTunes Store gekauft, so wird es automatisch zur lokalen Mediathek hinzugefügt.

**Software ohne Lebenszyklus**
Software in der Internet-Ära ist ein Service, das dauernd verfügbar ist, während herkömmliche, zu installierende Software ein Produkt ist. Dies wirkt sich natürlich auf den Software Entwicklungs- sowie Auslieferungsprozess aus. Der Entwicklungszyklus traditioneller Software kann stark vereinfacht in fünf Stufen gegliedert werden: Design, Implementierung, Test, Auslieferung und Installation. Die Entwicklung von Software im Web hingegen folgt dem Perpetual Beta Modell: Dabei befindet sich die Software in einem fortlaufenden Beta Stadium, ist also nie fertig. Stattdessen wird die Software ständig weiterentwickelt und verbessert. Änderungen werden dabei so oft wie möglich durchgeführt und die erneuerte Version im Web verfügbar gemacht[Musser and O'Reilly, 2006].

Die Benutzer spielen eine wichtige Rolle in diesem Entwicklungsprozess: Sie werden zu Co-Entwicklern und unverzichtbaren Testern. Ihr Feedback wird zur Kenntnis genommen und sofort eingearbeitet. Dies ist bei weitem wertvoller als Marketing Recherchen oder Tests von Prototypen durch eine geringe Zahl von Usern. Abgesehen davon schafft es eine größere Kundenbindung und auf die Vorstellungen und Wünsche der Benutzer wird eingegangen, sie gestalten die Software in gewissem Maße mit.

---

[26] `http://www.apple.com/itunes`

**Nutzen des "Long Tail"**

[Anderson, 2004] stellte die Theorie des Long Tail ("Der lange Schwanz") auf, in der eine Veränderung des Marktes und neue Möglichkeiten der Produktion und Distribution durch das Internet beobachtet wurde. Im Grunde besagt die Theorie, dass Unternehmen durch das Anbieten einer großen Anzahl an Nischenprodukten gewinnbringend wirtschaften können. Das Phänomen des Long Tail bezieht sich auf eine statistischen Verteilung, die oft zwischen der Popularität und der Anzahl von Produkten auftritt und dabei die Form eines langen Schwanzes hat (siehe Abbildung 2.1).

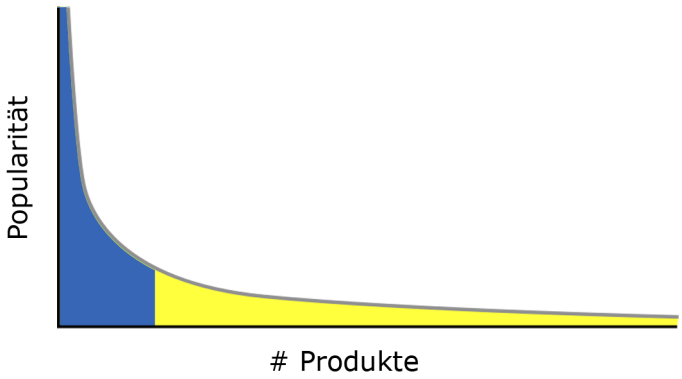



Abbildung 2.1: Das Modell des Long Tail, (Kurve entnommen aus Hay Kranen's Grafik `http://en.wikipedia.org/wiki/File:Long_tail.svg`)

Der linke Teil der Grafik (dunkler Bereich) repräsentiert die beliebtesten bzw. populärsten Produkte, also Bestseller am Kopf der Nachfragekurve. Dieser Kopf fällt sehr schnell ab und führt zu einem erheblich größeren Teil des Marktes (heller Bereich), dem langen Schwanz. Die Produkte dieses Bereiches befinden sich in Nischen und werden zwar weniger oft gekauft als die Hits am Beginn der Kurve, eröffnen jedoch durch ihr Nischendasein deutliche Marktchancen[Musser and O'Reilly, 2006]. So beobachtete [Anderson, 2004], dass der Online DVD Verleih Netflix[27] zwischen 35.000 und 40.000 verschiedene Filme pro Tag verleiht. Allerdings handelt es sich dabei nur bei ca. 3.000 Titeln um große Filmproduktionen (sog. "Blockbuster"). Dies bedeutet, dass der Großteil des Umsatzes auf unbekanntere Filme und Independent Produktionen fällt. Gleiches gilt für den Onlinemusikdienst Rhapsody[28], der zum damaligen Zeitpunkt in etwa 735.000 Titel im Angebot hatte. Nur ein geringer Teil des Umsatzes fällt hier auf die Top 10.000 Musikstücke, während unbekanntere Titel aus Nischenmärkten für die Mehrheit des Umsatzes verantwortlich waren.

[27] `http://www.netflix.com/`
[28] `http://www.rhapsody.com`

Das Internet ist aus verschiedenen Gründen geeignet, die Möglichkeiten des Long Tail zu nutzen. Zum einen gelten für reale Märkte Beschränkungen, die für das Internet nicht existieren. So braucht man zum Speichern von digitalen Medien keine Warenhäuser oder Lagerhallen, wie dies z.B. für Musik CDs oder Film DVDs notwendig ist. Der Großteil des Internets besteht aus kleinen Seiten, welche die Erschließung von Nischenmärkten ermöglichen. Am herkömmlichen Markt ist die Nachfrage nach Nischenprodukten durch gegraphische Beschränkungen zu gering, und somit die Kosten oft zu hoch. Weitere Kostenvorteile gegenüber realen Märkten hat der Onlinehandel in den Bereichen Vertrieb, Inventarisierung sowie Verkauf[Musser and O'Reilly, 2006].

**Schlanke und skalierbare Programmiermodelle**
Lautete der Slogan für Unternehmen des Web 1.0 noch "get big fast", so heißt er für das Web 2.0 "small is the new big"[Godin, 2006]. Dies bezieht sich auf Geschäftsmodelle sowie Technologien und Entwicklungsmodelle des Web 2.0: Diese sollten schlank, einfach und skalierbar zugleich sein, um Kosten und Risiken zu vermeiden. Um eine Anwendung erfolgreich umzusetzen und im Web zu starten, braucht es heute weder große Entwicklerteams, noch bedarf es riesiger Marketing Budgets. Virales Marketing und Netzwerkeffekte helfen einer Applikation, populär zu werden.

Preise für Hard- und Software sowie für Datenübertragungsbandbreiten sind durch die Kommerzialisierung um ein vielfaches gefallen. Open Source Software wie z.B. die Linux, Apache, MySql und PHP (LAMP) Web Plattform ist lizenzfrei und findet breiten Einsatz. In Kombination mit weiteren frei verfügbaren Modulen und Bibliotheken kann man mit kleinem Budget in kurzer Zeit akzeptable Lösungen entwickeln. Agile Entwicklungsprozesse und iterative Produktzyklen (vgl. 2.1.1.2) gepaart mit Kundeneinbindung schaffen Vertrauen, reduzieren Kosten, Zeit und Risiko. Durch die Einfachheit der Geschäfts- und Programmiermodelle kann man flexibel auf Marktbedürfnisse reagieren und so bei Bedarf schnell skalieren [Musser and O'Reilly, 2006]. Die Social News Webseite Digg.com demonstrierte kosteneffektive Skalierbarkeit vorbildlich: Digg startete mit einem Anfangsbudget von $2.000 und einem einzigen Entwickler. Es wurde Open Source Software eingesetzt und das Webhosting über einen Server um $99 durchgeführt. Etwa zwei Jahre später verfügte die Seite über mehr als 500.000 registrierte Benutzer und hatte 10 Millionen Seitenaufrufe pro Tag, verteilt auf ca. 90 Webserver. Dennoch war die Zahl der Entwickler nur auf 15 Personen gestiegen[Stern, 2006].

### 2.1.2 Strukturierte Informationen

Da in der vorliegenden Arbeit der Begriff "strukturierte Informationen" elementar ist, ist es notwendig, diesen eingehend zu definieren. Informationen im Internet können grundsätzlich in drei verschiedene Strukturierungsgrade eingeteilt werden: Unstrukturiert, semi-strukturiert und strukturiert (vgl. [Heuer and Priebe, 2000], [Abiteboul, 1997]).

Unstrukturierte Daten haben keinerlei Struktur und werden auch als "raw data" bezeichnet. Einerseits zählen Binärdateien wie Bilder und Musikdateien dazu, andererseits auch einfache Textdateien, die nur Fließ- bzw. Freitext und keinerlei sonstige Metadaten bzw. Auszeichnungen enthalten.

Im Gegensatz dazu spricht [Abiteboul, 1997] von strukturierten Daten, wenn die Struktur der Daten in einem Schema beschrieben ist und diese in einer Datenbank abgelegt sind. In relationalen Datenbanken werden Informationen in Tabellen gespeichert, die in Beziehung zueinander gesetzt werden. Die Tabellen liegen einem Datenmodell zugrunde, welches verwendet wird, um einen Teil der Wirklichkeit abzubilden. Die Struktur der Datenobjekte wird getrennt von den eigentlichen Informationen in einem Datenbankschema beschrieben [Kemper and Eickler, 2006]. Jede Tabelle besteht aus einer beliebigen Anzahl von Spalten, welche einen bestimmten Datentyp besitzen, z.B. Integer (Ganzzahl), Char (Zeichenkette) oder Blob (binäres Format) uvw. Die Spalten dienen der Beschreibung einer bestimmten Klasse von Objekten, sie sind also die Attribute. Z.B. könnte man zur Beschreibung einer Person je eine Spalte für den Namen, die Telefonnummer sowie die E-Mailadresse verwenden. Dabei würde der Name und die E-Mail den Datentyp Char erhalten und für die Telefonnummer würde Integer als Typ in Frage kommen. In der Arbeit selbst bezieht sich der Begriff "strukturierte Informationen" auf die vorhin erläuterte Definition: Informationen, die in einer relationalen Datenbank strukturiert in Tabellen abgelegt sind.

Zu guter letzt ist noch die Kategorie der semi-strukturierten Daten zu erwähnen. Semi-strukturierte Daten sind weder unstrukturiert, noch strikt typisiert, wie sie z.B. in Tabellen in relationalen Datenbanken gespeichert sind[Abiteboul, 1997]. Semistrukturierte Daten werden als selbstbeschreibend oder "Schema-los" bezeichnet. D.h., dass keine eigene Beschreibung der Struktur oder des Typs der Daten gebraucht wird. Stattdessen werden die Daten mittels einer einfachen Syntax beschrieben [Abiteboul et al., 2000]. Das folgende Beispiel zeigt ein semi-strukturiertes Datum von Label-Value-Paaren:

```
{name: "Hans", tel: 2748921, email: hans@dao.com}
```

Codebeispiel 2.1: semi-strukturiertes Label-Value-Paar

Im Gegensatz zu Informationen in Datenbanktabellen sind semi-strukturierte flexibler und können in beliebiger, nicht vorher festgelegter Struktur gespeichert werden: Label-Value-Paare können in Datensätzen doppelt vorkommen, ganz wegge-

lassen oder beliebig verschachtelt werden[Connolly and Begg, 2005]. Informationen, die im Datenaustauschformat XML[29] gespeichert sind, sind ebenfalls semi-strukturiert[Teorey et al., 2005]. In XML werden die Daten mittels Tags (Ausdrücke in spitzen Klammern) ausgezeichnet. Das oben genannte Beispiel könnte in XML-Notation so geschrieben werden:

```
<person>
  <name>Hans</name>
  <tel>2748921</tel>
  <email>hans@dao.com</email>
</person>
```

Codebeispiel 2.2: semi-strukturierte Daten in XML

Natürlich ist es auch möglich, semi-strukturierte Informationen z.B. in XML-Notation in einer Datenbanktabelle zu speichern. Obiges Beispiel in XML könnte in nur einer Spalte mit dem Datentyp Char gespeichert werden. Dies wäre in diesem Fall nicht sinnvoll, und i.d.R. wird man dies bei der Erstellung des Datenmodells vermeiden und stattdessen für jedes Attribut eine eigene Spalte verwenden.

Betrachtet man strukturierte und unstrukturierte Informationen aus Benutzerperspektive im Kontext von Web 2.0, so ist folgendes erkennbar:

- Benutzer sind bereit, Informationen im Web 2.0 zu erstellen und zu veröffentlichen.

- Sie bevorzugen allerdings die Verwendung von unstrukturierter Information gegenüber strukturierter, wie die Beispiele Wikipedia oder Tagging zeigen. Diese sind leicht einzugeben, aber von geringer Qualität und mit hohem Rauschen behaftet (vgl. [Schaffert et al., 2007]).

- Für akkurate Analysen ist jedoch strukturierte Information i.A. hilfreicher. Die Eingabe von strukturierten Informationen ist jedoch aufwendiger, weil sich User an vorgegebene Strukturen bzw. Konventionen halten müssen.

- Dies gilt genauso für den Anwendungsfall: Um exakte Rückschlüsse ziehen zu können, müssen strukturierte Daten verarbeitet werden. Nachdem auch die durch externe User erstellten Daten verwertet und analysiert werden sollen, müssen auch diese strukturiert vorliegen. Aus diesem Grund werden im kommenden Kapitel Plattformen zur Diskussion von strukturierten Informationen betrachtet.

---

[29] eXtensible Markup Language

## 2.2 Plattformen zur Diskussion von strukturierten Informationen

Im vorigen Kapitel wurden einige Definitionen angeführt, um Begriffe abzugrenzen und Klarheit zu schaffen. In diesem Abschnitt werden verschiedene Plattformen beschrieben, die vor allem im Web 2.0 zur Diskussion eingesetzt werden. Zu jeder Plattform wird eine Definition gegeben und ihre geschichtliche Entwicklung beleuchtet. Nach den Grundlagen wie typische Merkmale und charakteristische Funktionen werden die Möglichkeiten der Diskussion i.A. einer jeden Plattform aufgezeigt. Zu guter letzt wird das Potenzial der Diskussion für den Anwendungsfall jeder Plattform besprochen. Neben Blogs, die durch das Web 2.0 an Popularität gewannen, werden Wikis sowie Diskussionsforen behandelt. Ein Fazit schließt das Kapitel ab.

### 2.2.1 Blogs

#### 2.2.1.1 Definition

Ein Blog ist eine Webseite bzw. Online-Publikation, die regelmäßig aktualisiert wird und deren Beiträge i.d.R. umgekehrt chronologisch geordnet sind. Dabei ist der aktuelleste Beitrag oben auf der Startseite zu finden, gefolgt von älteren Beiträgen. Das Wort Blog ist eine Abkürzung des Begriffes Weblog, welches eine Zusammensetzung aus Web und Log ist (vgl. [Stocker and Tochtermann, 2009], [Alby, 2007], [Bartel, 2007]). Log bezieht sich in diesem Zusammenhang auf die einzelnen Einträge, die publiziert werden, angelehnt an die Einträge eines Logbuches in der Nautik. Ein Blog hat den Charakter eines Tagebuches oder Journals, das im Web veröffentlicht wird und dessen Einträge kommentiert werden können. So bezeichnet [Schmidt, 2008] Weblogs als “eine Kombination aus persönlichen Homepages und Diskussionsforen”.

#### 2.2.1.2 Geschichte

Weblogs existieren im Prinzip bereits seit Anbeginn des Webs. Nach [Möller, 2006] war bereits die erste verfügbare Website im World Wide Web, info.cern.ch, ein Blog. Tim Berners-Lee veröffentlichte darauf regelmäßig Links auf neue Seiten im Web. Der Begriff Weblog wurde von Jorn Barger 1997 geprägt [McCullagh and Broache, 2007]. Er bezeichnete damit den Prozess des “logging the web” beim Websurfen. Doch erst mit der Verfügbarkeit von einfach zu bedienenden Weblog-Publishing-Systemen wurden Blogs populär [Alby, 2007]. Ohne diese Software musste man über HTML-Kenntnisse verfügen und langwierige “Download-Bearbeiten-Upldoad-Szenarien” durchlaufen, um Inhalte im World Wide Web zu publizieren [Groß and Hülsbusch, 2004].

#### 2.2.1.3 Grundlagen

[Blood, 2002] unterscheidet Weblogs in drei Grundtypen inhaltlicher Natur: Filter-Blogs, persönliche Journale und Notebooks ("Notizbücher"). Während auf persönlichen Journalen privates sowie eigene Meinungen und Gefühle zum Ausdruck gebracht werden, beinhalten Filter-Blogs in erster Linie Links zu ausgewählten, publizierten Themen im Web, die man als interessant bzw. lesenswert empfindet. Die Beiträge auf Notebooks sind lange, fokussierte Aufsätze über privates als auch allgemeine Themen. Heute wird praktisch über jedes Thema gebloggt, was zu einer weiteren inhaltlichen Klassifizierung der Blogs führte: "Watchblogs" z.B. beobachten Unternehmen und Medien kritisch, "Litblogs" beschäftigen sich mit Literatur und "Blawgs" mit juristischen Themen. Neben Corporate Blogs, welche in Unternehmen als Instrument zur Kommunikation eingesetzt werden, sowie Fotoblogs, auf denen vor allem Fotos publiziert werden, entstanden noch viele weitere Arten von Blogs [Alby, 2007].

Blogs unterscheiden sich heute von "herkömmlichen" Webseiten durch verschiedene Merkmale und werden nach [Bartel, 2007] durch mindestens vier Eigenschaften gekennzeichnet:

- Chronologie: Das wichtigste Merkmal eines Blogs ist die chronologische Sortierung der Einträge[Alby, 2007]. Dabei ist jeder Eintrag mit dem Zeitpunkt der Veröffentlichung versehen. I.d.R. ist der aktuellste Eintrag der erste auf der Startseite.
- Aktualität: Das Gros an veröffentlichten Atrikeln auf Blogs beschäftigt sich mit aktuellen Ereignissen.
- Interaktion: Leser von Blogs können Beiträge in den meisten Fällen kommentieren und somit zum Thema Stellung nehmen. Das BILDblog[30] z.B. erlaubt keine Kommentare und wird von einigen daher nicht als "echtes" Blog bezeichnet [Alby, 2007].
- Internet-Bezug: Beiträge enthalten typischerweise Links zu weiterführenden Informationen oder anderen Fundstellen. Auch ist es üblich, auf andere Blogs zu verlinken. [Blood, 2002] meint dazu: "if you are not linking to your primary material when you refer to it [...], you are not keeping a weblog".

#### 2.2.1.4 Technische Merkmale

Blogs sind keine neue Technologie, sondern Content Management Systeme (kurz CMS) zur Verwaltung und Bearbeitung der veröffentlichten Inhalte [Back et al., 2008]. In CMS sind die Inhalte, wie Text, Grafiken, etc., getrennt vom Design in einer Datenbank gespeichert. Neue Beiträge können dabei einfach und direkt im Webbrowser

[30] `http://www.bildblog.de/`

verfasst und bestehende bearbeitet werden. Nicht nur der Inhalt kann mit gängiger Blog-Software wie WordPress[31] oder Movable Type[32] gepflegt werden, auch Design und Layout des Blogs können selbst bestimmt und angepasst werden [Bartel, 2007].

Jeder Beitrag auf einem Blog ist mit einem Permalink versehen. Dies ist die Webadresse (Uniform Resource Locator, kurz URL), unter der ein einzelner Eintrag permanent aufgerufen werden kann. Die Möglichkeit, auf einzelne Beiträge Bezug zu nehmen und diese direkt zu verlinken, hat sich positiv auf die Blogosphäre ausgewirkt. Der Ausdruck Blogosphäre wurde von William Quick ins Leben gerufen[33] und bezeichnet die Gesamtheit aller Blogs [Sauers, 2006]. Durch die gute Vernetzung der Blogs untereinander breiten sich neue Themen schnell aus (vgl. [Back et al., 2008], [Alby, 2007]).

Trackbacks sind ein weiteres technisches Merkmal, das Blogs auszeichnet und ebenfalls grundlegend zur Entwicklung der Blogosphäre beigetragen haben. Trackback ist ein Framework zur Kommunikation zwischen Websites bzw. Blogs. Die Firma Six Apart[34] entwickelte diese Funktion, um den Autor eines Blogs zu informieren, wenn auf dessen Beitrag in einem anderen Blog Bezug genommen wurde [Alby, 2007]. Setzt ein Blogger (Autor des Blogs) einen Trackback, so sendet die Blog-Software eine Nachricht (Ping) an den zu verlinkenden Blog. Die empfangende Blog-Software definiert die Beziehung zwischen Sender und Empfänger, indem sie ihrerseits einen Link zum Sender erstellt [SixApart, 2004].

Benutzer haben die Möglichkeit, auf Blogbeiträge über Feeds zuzugreifen. Ein Feed ist nach [Sauers, 2006] "a document (often XML-based) which contains content itmes, often summaries of stories or weblog posts with web links to longer versions". Technsich gesehen handelt es sich dabei um ein Datenformat, das von Programmen, sog. Feedreader oder Aggregatoren, gelesen und aufbereitet wird und nicht für Menschen bestimmt ist. Feeds werden nicht nur von Blogs sondern auch z.B. von News-Websites eingesetzt, um Inhalte an interessierte Nutzer zu verteilen. Durch Blogs haben Feeds allerdings an Popularität gewonnen [Alby, 2007]. Feeds bringen einige Vorteile mit sich: Nutzer müssen eine Webseite nicht aufrufen, um zu Informationen zu gelangen. Stattdessen werden sie automatisch über neue Inhalte informiert. Feeds folgen dem Pull-Prinzip, im Gegensatz zu E-Mails, welche gepusht werden. D.h., dass Nutzer nur jene Inhalte erhalten, welche sie abonnieren. Inhalte können nach den Vorlieben des Lesers zusammengestellt werden, wodurch er uninteressante Informationen gar nicht bekommt. Dabei ist die Angabe der E-Mail-Adresse nicht notwendig, was unangenehme Spam-Attacken vermeidet. Ist ein Ende des Informationsbezuges erwünscht, so bestellt der Leser den Feed einfach ab [Back et al., 2008].

---

[31] `http://wordpress.org/`
[32] `http://movabletype.org/`
[33] `http://dailypundit.com/?p=10823`
[34] `http://www.sixapart.com/`

Es existieren verschiedene Standards für Feeds, wobei RSS 2.0 das am weitest verbreitete Format ist. RSS ist eine Abkürzung für "Really Simple Syndication"[35], also "ganz einfache Verteilung von Inhalten" [Bartel, 2007]. RSS ist nicht nur ein Format zur Verbreitung von Nachrichten, es kann auch verschiedene multimediale Inhalte wie z.B. Podcasts abbilden. Aktuell existiert RSS in der Version 2.0 und wurde seit 1999 von verschiedenen Entwicklergruppen teilweise parallel weiterentwickelt. 2003 wurde das Verbreitungsformat Atom ins Leben gerufen, welches die Nachfolge von RSS antreten und es als vorherrschendes Verbreitungsformat ablösen will. Atom versucht die Vorteile der verschiedenen Standards zu vereinen [Back et al., 2008]. Es wurde im Gegensatz zu anderen Formaten vor allem auf Basis der Anforderungen von Bloggern entwickelt. So wird z.B. Mehrsprachigkeit und damit verbunden das Einbinden von nicht lateinischen Buchstaben in Feeds unterstützt [Sauers, 2006].

Zur thematischen Gliederung der Blogbeiträge können diese einer oder mehreren Kategorien zugeordnet werden. Beiträge lassen sich nach diesen Kategorien filtern, wodurch Blog-Leser gezielt das lesen können, was sie interessiert (vgl. [Kaiser, 2008], [Bartel, 2007]).

#### 2.2.1.5 Potenzial der Diskussion i.A.

Technorati[36], Betreiber einer der größten Suchmaschinen für Weblogs, führt jedes Jahr eine Untersuchung über die Blogosphäre durch und veröffentlicht einen Bericht über den gegenwärtigen Status. Aus dem aktuellen Bericht (vgl. [Sussman, 2009]) geht hervor, dass Blogger mittlerweile in vier verschiedene Gruppen eingeteilt werden können: Hobby-Blogger sind mit einem Anteil von 72% nach wie vor die größte Gruppe. Danach folgen Teilzeit-Blogger (15%), welche Ihr Einkommen mit dem Bloggen aufbessern, Selbstständige (9%) und Unternehmensblogger (4%). Selbstständige sind die professionellsten Blogger, bloggen für das eigene Unternehmen bzw. sehen den Blog zum Teil "als ihr Unternehmen" bzw. eigentliche Beschäftigung an. Bei der Betrachtung der demographischen Verteilung ist zu erkennen, dass ca. die Hälfte aller Blogger aus den Vereinigten Staaten und in etwa ein Viertel aus Europa stammen.

Nutzer bloggen nicht nur, um über ihr Leben zu berichten, ihre Meinungen und Gefühle auszudrücken, sondern auch um andere Autoren und Leser partizipieren zu lassen. Sie wollen, dass andere Leser an ihren Gedanken teilhaben, ihre Ansichten preisgeben und ihnen Feedback zu ihren Publikationen geben [Nardi et al., 2004]. Mittlerweile ist auch die monetäre Vergütung eine der Gründe für das Bloggen [Sussman, 2009]. Nach [Gumbrecht, 2004] und [Trevino, 2005] legen Blogger auch großen Wert auf Feedback und sehen Kommentare und damit die Diskussion als

[35] In den früheren Versionen steht RSS für "Rich Site Summary" (Versionen 0.9, 0.91, 0.92) bzw. "RDF Site Summary" (Vresion 1.0) [Back et al., 2008]

[36] `http://technorati.com`

integralen Bestandteil des Bloggens.

Die Diskussion auf Blogs erfolgt also in erster Linie über die Kommentarfunktion. Kommentare werden am Ende jedes Blog-Posts hinzugefügt und untereinander in einer chronologisch sortierten Liste (Thread, Diskussionsstrang) angezeigt. Gängige Blog-Publishing-Systeme bzw. Blog Provider wie Blogger[37] oder WordPress[38] bieten vielseitige Einstellungen und nützliche Funktionen an, um die Diskussion möglichst einfach und nachvollziehbar zu gestalten. User können dabei regeln, ob sie Kommentare grundsätzlich erlauben oder verbieten, wie Kommentare angezeigt werden (chronlogisch sortiert oder neuester Beitrag zuerst), ob Benutzer registriert sein müssen oder jeder das Recht hat, Kommentare abzugeben uvm.

Neben grundlegenden Einstellungen können Blogger auch festlegen, ob sie E-Mail Benachrichtigungen erhalten wollen, wenn neue Kommentare hinzugefügt wurden, um auf dem Laufenden zu bleiben und dazu nicht immer den Blog besuchen zu müssen. Mittels der Blog-Software WordPress können neben Blog-Beiträgen auch der Kommentarthread eines Posts als Feed abonniert werden. So muss nicht der Blog selbst besucht, sondern kann ein Newsreader verwendet werden, um die Diskussion zu verfolgen [WordPress, 2009].

Weiters ist meist eine Einstellung zur Moderation von Kommentaren verfügbar, um abfällige Kommentare sowie unerwünschte Spam-Kommentare zu vermeiden. Moderation bedeutet, dass Kommentare nicht sofort am Blog-Post erscheinen, sondern zuvor noch vom Blogger freigegeben werden müssen. Dies wird über eine Webschnittstelle realisiert, über die Kommentare gelöscht, freigegeben, als Spam gemeldet oder auch bearbeitet werden können. Desweiteren kann auf bestimmten Blogging-Systemen direkt über dieses Interface eine Antwort auf Kommentare gegeben werden, welche dann auch im Kommentarthread zum jeweiligen Post angezeigt wird. Das direkte Antworten auf Kommentare selbst ist nicht nur dem Autor selbst vorbehalten, sondern ist jedem User gestattet. Dadurch kann eine Diskussion gezielter geführt und Kommentatoren direkt adressiert werden.

Blog-Kommentare besitzen ebenso wie Blog-Posts einen Permalink. So ist es möglich, nicht nur den gesamten Blog-Beitrag, sondern nur einen Teil der Diskussion am eigenen Blog zu adressieren. Dies kann allerdings die gleichen Folgen haben, wie Trackbacks, das im Folgenden behandelt wird. Mit der Trackback-Funktion, wird ein Blogger informiert, wenn ein anderer Autor dessen Beitrag verlinkt hat (vgl. 2.2.1.4). Wurde ein Trackback auf einen Blog-Eintrag gesetzt, so erscheint dieser im referenzierten Post entweder ober- oder unterhalb der Kommentare in der Form eines Links auf den referenzierenden Artikel. Somit fördern Trackbacks die Disukussion, auch wenn sie dadurch in ein anderes Umfeld verschoben wird, nämlich auf den

[37] `https://www.blogger.com/`
[38] `http://wordpress.com/`

Blog des zitierenden Autors [Alby, 2007]. [de Moor and Efimova, 2004] führen Blog Kommentare und Diskussionen in den Kontext von Konversationen. Dabei gewinnen sie unter anderem die Erkenntnis, dass Konversationen zu einem Thema über mehrere Blogs verteilt geschehen:

> *"One of my constant frustrations is not being able to keep track of a conversation when it's spread across weblogs and comments on weblogs."*

Kommentare werden also einerseits direkt im dafür vorgesehenen Kommentarbereich des verfassten Artikels abgegeben, andererseits auch auf Antwortbeiträgen anderer Blogger. Hinzu kommt, dass Blogger auch über andere Kommunikationskanäle wie Instant Messaging mittels Skype[39] oder ICQ[40] kommunizieren. Dabei geht ein Teil der Diskussion verloren bzw. wird nicht dokumentiert. Dem Problem von Kommentaren verteilt auf mehrere Blogs versuchen Web-Kommentarsysteme entgegenzukommen, welche im nächsten Abschnitt näher betrachtet werden.

**Web Kommentarsysteme**

Web Kommentarsysteme bzw. -services versuchen der Fragmentierung von Kommentaren bzw. Diskussionen über mehrere Quellen (z.B. Blogs, CMS) Einhalt zu gebieten und eine effektivere Diskussion im Web zu ermöglichen. Die drei zur Zeit bekanntesten Kommentarservices sind Disqus[41], IntenseDebate[42] und Echo[43] [Bansal, 2009]. Die Systeme funktionieren alle nach einem ähnlichen Prinzip und können bei selbst-gehosteten Blogsystemen als Plugin installiert oder als JavaScript eingebunden werden. Das Plugin sendet die Kommentare des eigenen Blogs an das Webservice, welches die Kommentare in einer zentral verwalteten Datenbank als "discussions" speichert. Die Kommentare werden dabei mit dem Kommentarsystem des Blogs synchronisiert. Durch das installierte Plugin ändert sich auch das User Interface, welches weitere Funktionen und eine übersichtlichere Darstellung (Verschachtelung) der Kommentare bietet. [Jangro, 2008].

Alle Systeme unterstützen u.a. das Antworten auf und Moderieren von Kommentaren per E-Mail, Anzeige der Trackbacks, das Abonnieren eines RSS-Feeds für Kommentare, die Bewertung von Kommentaren sowie das Synchronisieren der Kommentare zwischen dem Blog und dem Kommentarsystem. Zusätzlich werden die Kommentare in Echtzeit aktualisiert und am Blog sofort dargestellt, ohne die Seite neu laden zu müssen. Für eine vollständige Liste aller Features sei an dieser Stelle auf die Websites der Anbieter verwiesen. Die Systeme stellen nicht nur weitere nützliche Funktionen für das Kommentieren selbst zur Verfügung, vielmehr sind sie Services und zentrale Anlaufstelle zur Verwaltung der Kommentare von einem Punkt aus.

---

[39] `http://www.skype.com/`
[40] `http://www.icq.com/`
[41] `http://disqus.com/`
[42] `http://intensedebate.com/`
[43] `http://js-kit.com/`

Nach erfolgreicher Registrierung am Kommentarsystem wird ein Benutzerprofil erstellt. Abgesehen von Standard-Einstellungen wie Name, Username, E-Mail sowie Buddy-Icon, hat der Benutzer die Möglichkeit, sich mit dieser Identität auf anderen Blogs (die ebenfalls das gleiche Kommentarsystem installiert haben) einzuloggen und über das Service zu kommentieren. Auf allen drei Systemen können die User die Kommentare des oder der eigenen Blogs einsehen und verwalten. Diese werden aggregiert in einer Liste dargestellt. Auch auf die Antworten anderer User auf eigene Kommentare sowie über das System abonnierte RSS-Feeds kann vom System aus zugegriffen werden. Das Kommentarservices Disqus erlaubt umfangreiche Verwaltungsmöglichkeiten wie nachträgliches Bearbeiten, Löschen und Antworten auf Kommentare.

Alle drei Kommentarsysteme integrieren Web 2.0 Dienste. Das Login zum Kommentarsystem über Drittanbieter, wie z.B. Facebook, Twitter, OpenID[44] oder FriendFeed[45], unterstützt jedes System. Auch die Aggregation von sog. "social comments" wird von jedem System in unterschiedlicher Breite angeboten. Social comments sind Reaktionen bzw. Kommentare auf Benutzer generierten Inhalt, die von Benutzern in sozialen Netzwerken und Medienseiten wie z.B. Digg, YouTube oder Flickr abgegeben werden [Matsak, 2009]. IntenseDebate bindet bis dato nur Kommentare der FriendFeed Plattform ein, die in weiterer Folge auf dem Blog mit installiertem IntenseDebate Plugin auch im Kommentarstrang veröffentlicht werden [Koenig, 2009]. Das Echo Kommentarsystem hat eine ähnliche Funktion und bringt diese sozialen Kommentare als "Echo Stream" auf den eigenen Blog[46]. Dies geschieht durch Filterung der Kommentare und Meldungen innerhalb der Sozialen Netzwerke und Medienseiten, welche einen URL auf den Blog bzw. ein Post beinhalten [Cailloux, 2009]. Auch Disqus bietet diesen Service an und nennt ihn "Social Media Reactions". Realisiert wird es über die Konversations-Suchmaschinen Backtype[47] und uberVU[48] [Disqus, 2009]. Diese "Reactions" werden zwecks Übersichtlichkeit unter den direkten zum Artikel geposteten Kommentaren angezeigt. Zusätzlich bietet Disqus neben der Anzeige aller aggregierten (sozialen) Kommentare auch die Möglichkeit, diese zentral über die Plattform zu bearbeiten und zu löschen.

Die drei erwähnten Web-Kommentarsysteme haben alle das Ziel gemein, dem Problem der Fragmentierung von Konversationen im sozialen Web entegenzuwirken, indem Konversationen an einer zentralen Stelle konzentriert werden, um so eine effektivere Diskussion zu ermöglichen. Doch diese Aggregation kann sich auch negativ auf die Diskussion auswirken, wie [Singer, 2009] anmerkt:

---

[44] `http://openid.net/`

[45] `http://friendfeed.com/`

[46] Eingebunden werden zur Zeit Kommentare folgender Plattformen: Twitter, Facebook, Delicious, FriendFeed, Google Reader.

[47] `http://www.backtype.com/`

[48] `http://www.ubervu.com/`

> *"Aggregating 100's of people ReTweeting the subject line of your message underneath the conversation adds nothing to your community, except destroying the actual conversation your community is trying to have on that page itself. By aggregating noise from around the social web on your blog [...] you may be limiting the depth and impact of conversations."*

Das ReTweeting bezeichnet das referenzierte Wiederholen eines Beitrages anderer Benutzer am Micro-Blog Twitter. Wenn diese Kommentare (mit nahezu identischem Inhalt) durch ein Kommentarsystem aggregiert werden und am eigenen Blog in der Diskussion unter dem Blog-Eintrag angezeigt werden, so bereichert es nicht die Diskussion. Stattdessen mindert es die Qualität der Diskussion, führt zu einer Informationsüberflutung und einem Dickicht an Kommentaren ohne wirklichen Informationsgehalt bzw. Mehrwert. Lesenswerte Kommentare müssen dabei langwierig herausgefiltert werden. Nach [Mullenweg, 2009] sind es nicht nur ReTweets, sondern auch Links von Delicious sowie Kommentare von Digg und Slashdot[49], die eine Community vergraulen können. Diesem Problem kann auf einfache Art und Weise entgegengetreten werden. Durch eine Unterteilung in Kategorien von Kommentaren kann eine größere Übersicht geschaffen werden. So können diese Kommentare und Reaktionen aggregiert aus Social Media Sites unter den eigentlichen, direkten Kommentaren angezeigt werden. Oder die Ansicht der Kommentare wird in sog. Tabs bzw. Reiter unterteilt: Hier können User wählen, ob sie Reaktionen, tatsächliche Kommentare oder Trackbacks sehen wollen. Die dritte Lösung schlägt vor, die Kommentare aus dem sozialen Web gar nicht einzubinden und stattdessen nur direkt getätigte Kommentare mit Mehrwert anzuzeigen [Singer, 2009].

#### 2.2.1.6 Potenzial der Diskussion für den Anwendungsfall

Blogs sind i.d.R. bestens geeignet, um Micro-Contents zu verfassen, zu publizieren und diese zu disuktieren [Burg, 2003]. Die direkte Ansprache und Vernetzung von Blog-Posts ist zwar durch Permalinks gegeben, bei der Diskussion von strukturiertem Content im vorliegenden Anwendungsfall stoßen Blogs allerdings an ihre Grenzen. Gegenstand der Betrachtung sind Erfolgsprofile, die aus Erfolgsfaktoren und ihren Beziehungen zueinander bestehen. Ein Zugang wäre es, die Diskussion auf Ebene von Erfolgsfaktoren zu führen, also für jeden Erfolgsfaktor einen Blog-Beitrag zu erstellen. Dabei treten allerdings diverse Probleme auf: Bereits das Modellieren eines Problems bzw. Erfolgsprofils bringt schon Hürden mit sich. Einerseits bestehen Erfolgsprofile oft aus 100 oder sogar mehr Erfolgsfaktoren, es müssen also 100 Blog-Beiträge mit dem Namen des Erfolgsfaktors als Überschrift erstellt werden. Der eigentliche Inhalt könnte der Kurzbeschreibung des Faktors dienen, was vermutlich in den seltensten Fällen Sinn machen wird, da die Namen i.a.R. kurz, prägnant und selbstbeschreibend sind, wie z.B. "Mitarbeitermotivation" oder "Lernkultur schaffen". Zusätzliche Attribute wie Charakter (Stärke oder Schwäche) sowie etwaige

[49] `http://slashdot.org/`

Kategorien müssten ebenfalls direkt im Inhalt Platz finden. Verbindungen zu anderen Erfolgsfaktoren können über Links innerhalb des eigentlichen Blog-Posts sowie Trackbacks gelöst werden. Die Zuordnung eines Erfolgsfaktors zu einem Erfolgsprofil kann über die Kategorien erfolgen, ein Erfolgsprofil ist also genau einer Kategorie zugeteilt. Nachdem Kommentare zu Blog-Beiträgen gemacht werden, wird die Diskussion jedoch sehr unzufriedenstellend sein. Denn Kommentare werden genau zu einem Beitrag abgegeben, der nur aus einer Überschrift besteht und somit aus dem Kontext des gesamten Erfolgsprofils gerissen scheint.

Im zweiten Zugang wird die Diskussion auf Ebene des Erfolgsprofils geführt. Dabei formuliert ein Benutzer einen Teilaspekt seines Erfolgsprofils bzw. seines Problems auf einem Blog-Post inklusive der Erfolgsfaktoren sowie den Verbindungen möglichst detailliert. Schwierigkeiten treten hierbei bereits auf, welche Teile des Inhalts als Erfolgsfaktoren in Frage kommen und wie Beziehungen beschrieben werden. Die Diskussion selbst kann zwar Feedback über das Problem i.A. geben, jedoch wird es schwer sein, bestimmte Faktoren im engeren zu adressieren. In beiden Herangehensweisen ist es problematisch, dass Kommentare selbst wieder unstruktierten Inhalt beinhalten und damit schwer möglich, diese einem bestimmten Erfolgsfaktor zuzuordnen. Eine automatische Aggregation und akkurate Auswertung der Information (Erfolgsfaktoren und -verbindungen) aus Kommentaren von Blog-Posts ist zudem schwierig und aufwendig. Hier müssen Homonyme und Synonyme, unterschiedliche Schreibweisen und Abkürzungen, Umgangssprache sowie unvollständige Sätze bzw. fehlende Grammatik beachtet werden. Zudem sind die Kommentare auf verschiedene Blog Posts verteilt. Vor allem eine genaue Auswertung der Informationen hinsichtlich neuer Erkenntnisse zu Erfolgsfaktoren ist sehr problematisch und schwierig zu bewerkstelligen. Aufgrund der beschriebenen Probleme sind Blogs zur Diskussion von strukturierten Informationen im vorliegenden Anwendungsfall wenig geeignet.

### 2.2.2 Wikis

#### 2.2.2.1 Definition

Der Softwareautor Ward Cunningham konzipierte das erste wirkliche Wiki und definiert Wiki in [Leuf and Cunningham, 2001] wie folgt:

> *"A wiki is a freely expandable collection of interlinked Web "pages", a hypertext system for storing and modifying information - a database, where each page is easily editable by any user with a forms-capable Web browser client."*

Auch [Ebersbach et al., 2008a] definiert Wiki ähnlich und erweitert es noch um die Schlagwörter "Plattform" und "kooperatives Arbeiten":

> *"Ein Wiki ist eine webbasierte Software, die es allen Betrachtern einer Seite erlaubt, den Inhalt zu ändern, indem sie diese Seite online im Browser editieren. Damit ist das Wiki eine einfache und leicht zu bedienende Plattform für kooperatives Arbeiten an Texten und Hypertexten."*

Der Name Wiki eines solchen Softwaresystems leitet sich vom hawaiianischen Wort "Wiki" ab, was "schnell", "rasch" oder auch "sich beeilen" heißt. Der Begriff Wiki wird jedoch nicht nur für die Software selbst verwendet, sondern auch für das Konzept, das in Abschnitt 2.2.2.3 näher beschrieben wird (vgl. [Ebersbach et al., 2008a], [Leuf and Cunningham, 2001]).

#### 2.2.2.2 Geschichte

Die Grundidee stammt von Tim Berners-Lee, der das Web offen gestalten und Hypertextseiten für jedermann online bearbeitbar machen wollte [Bendel, 2006]. Der Softwareentwickler Ward Cunningham entwickelte das erste Wiki mit dem Namen WikiWikiWeb 1994 mit dem Ziel, Informationen über Software Entwurfsmuster schnell und kollaborativ zu veröffentlichen, sowie die Bearbeitungsschritte zu dokumentieren. Cunningham setzte sein System erfolgreich zur Dokumentation im Bereich der Softwareentwicklung und Entwurfsmuster ein (vgl. [C2.com, 2009c], [Leuf and Cunningham, 2001], [Ebersbach et al., 2008a]). Populär wurden Wikis erst mit der Entwicklung der freien Online-Enzyklopädie Wikipedia, dem größten und erfolgreichsten Wiki sowie Enzyklopädie weltweit. Wikipedia hat ihre Wurzeln im Vorgängerprojekt Nupedia, welches jedoch aufgrund der strikten Qualitätskontrolle der Artikel samt langwieriger Peer-Reviews durch Experten scheiterte und so nur sehr wenige Artikel veröffentlichte. Aus diesem Grund startete Jimmy Wales 2001 das Wikipedia-Projekt, wobei von technologischer Seite ein Wiki-System und somit auch das Konzept Wiki zum Einsatz kamen - jeder User darf jeden Artikel ändern, die Community reguliert sich selbst, Experten Reviews waren nicht mehr notwendig [Möller, 2003].

#### 2.2.2.3 Das Wiki Konzept

Der Grundgedanke von Wikis kann nach [Leuf and Cunningham, 2001] in folgenden Merkmalen zusammengefasst werden, erweitert und zum Teil ergänzt durch [Ebersbach et al., 2008a]:

- Ein Wiki hält jeden User an, jede beliebige Seite innerhalb des Systems zu bearbeiten sowie neue Seiten hinzuzufügen. Einzige Voraussetzung dafür ist ein herkömmlicher Browser, der gänzlich ohne Installation zusätzlicher Plugins oder anderer Software auskommt.

- Wikis kommen i.d.R. ohne hierarchische Navigationstrukturen aus. Durch intuitive und einfache Art Seiten zu verlinken, schaffen Wikis die Entstehung sinnvoller, assoziativer Hypertexte (vgl. [Ebersbach et al., 2008a]).

- Wikis sind keine vollständigen, fertigen Webseiten, sondern versuchen die Benutzer ständig zum Mitmachen, Kollaborieren und Erstellen von weiteren Seiten zu bewegen, um so die Webseite ununterbrochen zu verändern.
- Wikis schaffen einen Raum für Zusammenarbeit im Web, auch wenn dies aufgrund völliger Freiheit, einfacher Handhabung sowie simpler, ungewöhnlicher Navigationsstrukturen und mangels fixer, formaler Seitenstrukturen auf den ersten Blick nicht so scheint. Nicht die Technik steht im Vordergrund, vielmehr sind es die Entwicklung von Communities und Projekten, die Arbeitsprozesse, Diskussionen über inhaltliche Probleme sowie die sozialen Zusammenhänge einens Projekt, die in den Blick geraten (vgl. [Ebersbach et al., 2008a]).
- Wikis sind durch und durch demokratisch, denn alle Benutzer haben die gleichen Rechte, wodurch Webkollaboration auf einfache Weise und ohne Benutzerkonten ermöglicht wird.
- Die Eintrittsbarrieren zur Nutzung von Wikis und damit einhergehend der Kommunikation und Gestaltung des Webs sind aufgrund der Einfachheit der Software sehr niedrig. Es sind kaum technische Vorkenntnisse von Nöten, die Texteingabe erfolgt meist mit wenigen einfachen, schnell zu erlernenden Regeln [Ebersbach et al., 2008a].

Die wichtigsten Punkte des Wiki Konzepts wurden somit vorgestellt. Nicht nur die Wikipedia ist ein Paradebeispiel für ein erfolgreiches, funkionierendes Wiki. Die Gründe, warum Wikis funktionieren, hat Cunningham anhand seines Wikis[50] erörtert. Diese werden nachfolgend aufgelistet [C2.com, 2009a]:

- Die Gemeinschaft der Benutzer eines Wiki (Wiki Community) fühlt sich i.a.R. verpflichtet, den Inhalt des Wikis zu pflegen, denn jeder verwendet es.
- Wikis verkörpern die einfachste Form des Hyperlink Konzepts.
- Jede Information kann von jedem Benutzer geändert oder gelöscht sowie kommentiert werden. Wiki Seiten entstehen auf Basis von Diskussion und Konsens vieler. Unerwünschtes wird einfach gelöscht.
- Jeder darf das System ausprobieren und damit experimentieren. Auch wenn Wikis missbraucht werden und Vandalismus zum Opfer fallen, so entsteht grundsätzlich kein permanenter Schaden.
- Wikis sind keine WYSIWYG-Editoren[51]. Das Editieren von Wiki-Seiten ist zwar keine Wissenschaft, aber auch nicht trivial. Benutzer, die sich davon nicht angezogen fühlen, werden auch nicht an einem Wiki partizipieren. Übrig bleibt eine durch rationalen Diskurs charakterisierte Community.

---

[50] `http://c2.com/cgi/wiki`

[51] WYSIWYG ist ein Akronym für "What You See Is What You Get"

- Wikis sind keine Echtzeit-Systeme. Häufig haben und nehmen sich User Zeit, um Artikel zu verfassen, was oft zu sinnvollen und wohldurchdachten Inhalten führt.
- Wiki-Benutzer sind oft pedantisch, überkritisch und Querdenker. Dadurch entwickelt sich ein besonderes Gemeinschaftsgefühl.
- Wikis sind von Natur aus unstrukturiert. Für viele ist es reizvoll, die Freiheit zu haben, Seiten nach eigenen Vorstellungen zu erstellen und zu verlinken, anstatt eine Struktur diktiert zu bekommen.

#### 2.2.2.4 Technische Merkmale

Nachdem Ward Cunningham sein WikiWikiWeb-System erfolgreich einsetzte, dauerte es nicht lange, bis erste Nachahmer weitere Wiki-Systeme (auch Wiki Engines oder Klone genannt) entwickelten. Es war nicht schwer, den Code anzupassen oder auf andere Programmiersprachen zu portieren [Leuf and Cunningham, 2001]. So zählt [WikiMatrix, 2009a] heute mehr als 120[52] verschiedene Wiki Engines in nahezu jeder Programmiersprache. Zu den bekanntesten zählen die PHP Engines MediaWiki[53] (von Wikipedia eingesetzt) und DokuWiki[54], das in Perl geschriebene TWiki[55] sowie das JSPWiki[56] als Vertreter der Programmiersprache Java [WikiMatrix, 2009a]. Dabei ist anzumerken, dass die überwiegende Mehrheit der Wiki Systeme als Open Source Software verfügbar ist, ganz dem Gedanken von Wiki entsprechend.

Nicht nur viele verschiedene Wiki Engines sind verfügbar, nach [West, 2008] gibt es auch drei verschiedene Kategorien von Wiki Typen, was Installation und Verwendung betrifft: Freie und kostenpflichtige Wiki Services sowie selbst gehostete Wikis. Jeder Typ hat seine Vor- und Nachteile, die nun besprochen werden. Freie Wiki Services werden von kostenfreien Providern wie Wetpaint[57] oder Wiki Spot[58] angeboten. Dabei ist keine Installation von Software notwendig, da das Wiki auf einem Server des Providers betrieben wird. Die Einrichtung sowie Administration ist meist einfach und wird über einen Browser bewerkstelligt. Dies gilt auch für kostenpflichtige Wiki Services. Oft werden bei freien Wikis die Anzahl der Nutzer oder angelegten Seiten sowie Speicherplatz für die Inhalte beschränkt, was bei kostenpflichtigen Services kaum zutrifft. Kostenpflichtige Wiki Services bieten zusätzliche Funktionen, erweiterte Administrationsmöglichkeiten sowie mehr Sicherheit. Selbst gehostete Wikis müssen auf einem Server installiert werden, erfordern also technisches Know-How. Allerdings hat man dadurch die alleinige Kontrolle was Zugriff,

[52] [C2.com, 2009b] listet sogar 148 verschiedene Wiki Engines auf.
[53] http://www.mediawiki.org/
[54] http://www.dokuwiki.org/
[55] http://twiki.org/
[56] http://www.jspwiki.org/
[57] http://www.wetpaint.com/
[58] http://wikispot.org/

Benutzerrechte sowie Sicherheit anbelangt. Zudem hat man i.d.R. größere Speicherkapazitäten, muss aber einen eigenen Server betreiben bzw. Zugang dazu haben. Abgesehen davon dauert die Inbetriebnahme länger als wenn ein Wiki als Service in Anspruch genommen wird.

Technisch gesehen sind Wikis Software-Systeme mit einer Client-Server Architektur, die im Web betrieben werden und auf der Grundlage von einfachen Content-Management-Systemen funktionieren (vgl. [Leuf and Cunningham, 2001], [Bendel, 2006]). Die Seiteninhalte werden nicht in der Seitenbeschreibungssprache HTML, sondern mittels eines einfachen Wiki-Markups verfasst und in einer Datenbank oder in Dateien gespeichert. Beim Zugriff auf eine Seite im Wiki über den Browser wird der Wiki-Text durch die Wiki Engine in HTML übersetzt und an den Benutzer gesendet [Ebersbach et al., 2008a]. Um Text lesbarer zu gestalten und zu gliedern, wird eine Wiki-Syntax verwendet. Diese ist die Gesamtheit der möglichen Zeichenkombinationen. Diese unterscheidet sich je nach Wiki Engine. Die verschiedenen Wiki-Syntax Dialekte sind einfacher zu verstehen und zu schreiben als HTML, bieten jedoch ähnliche Möglichkeiten wie HTML [Koch and Richter, 2008]. Ein Beispiel soll dies demonstrieren. Um in HTML einen Link auf die Adresse der Wikipedia zu setzen, muss folgendes geschrieben werden:

```
<a href="http://de.wikipedia.org/">Wikipedia</a>
```

Codebeispiel 2.3: HTML Syntax

In der Wiki-Syntax von MediaWiki wird das gleiche Resultat durch folgende Auszeichnung erzielt:

```
[http://de.wikipedia.org/ Wikipedia]
```

Codebeispiel 2.4: MediaWiki Syntax

In dieser Syntax reicht es also, den Link zusammen mit dem Text auf den Link getrennt durch ein Leerzeichen innerhalb eckiger Klammern zu setzen.

#### 2.2.2.5 Charakteristische Funktionen

Unabhängig von verwendeter Wiki-Engine bietet jedes Wiki einige charakteristische Funktionen, die im Folgenden besprochen werden. Diese sind [Ebersbach et al., 2008a] entnommen. Wie bereits im vorigen Abschnitt erwähnt, kann jeder Benutzer jede Wiki-Seite einfach ändern. Dies ist das typischste Merkmal eines Wikis und wird meist über einen “Seite bearbeiten”-Link realisiert. Über ein Eingabefenster kann der Text der Seite in Wiki-Markup bearbeitet werden. Werden durchgeführte Änderungen gespeichert, so wird eine neue Version der Seite angelegt, ohne die alte zu überschreiben. D.h. es wird zu jeder Seite eine sog. “History” aller Versionen abgelegt. Durch diesen Mechanismus kann jederzeit auf eine ältere Version eines Artikels zugegriffen und diese bei Bedarf wiederhergestellt werden. So kann vandalierenden Usern entgegengewirkt werden. Gewisse Wiki-Engines verfügen über eine

Diff-Funktion. Diese stellt zwei Versionen eines Artikels nebeneinander dar, um die Unterschiede auf einen Blick zu erkennen.

Links sind ein weiteres wichtiges Merkmal von Wikis. Jeder Artikel kann auf jeden anderen verlinken. Dies geschieht je nach Wiki-Engine anders. Z.B. werden sie bei MediaWiki in eckige Klammern geschrieben, viele Wikis unterstützen aber auch sog. Wiki-Words. Ein Wiki-Word beginnt mit einem großen Anfangsbuchstaben, alle weiteren Wörter werden ohne Zwischenraum ebenfalls mit einem Großbuchstaben angehängt, also z.B.: "WordlWideWeb" oder "WikiEngine". Existiert eine Seite zu einem Link innerhalb des Wikis noch nicht, so kann diese mit einem Mausklick auf den Link erzeugt werden. Dadurch werden assoziative Verbindungen zwischen den Seiten unterstützt.

Typisch für Wikis ist auch die Funktion "Recent Changes", welche eine Übersicht von kürzlich geänderten Seiten (innerhalb eines Zeitraumes oder eine bestimmte Anzahl) gibt. Die Liste wird automatisch erstellt und enthält Links zu den geänderten Artikeln sowie zum Benutzer, der diese durchgeführt hat. User bleiben mit Hilfe der Funktion über Änderungen im Bilde, was vor allem Sinn bei einem Wiki mit einer Qualitätskontrolle der Inhalte macht. Weitere häufig implementierte Wiki-Funktionen sind die Sandbox sowie die Suche. In einer Sandbox ("Sandkasten") können v.a. Einsteiger die Wiki-Syntax ausprobieren und die Umgebung kennenlernen, um sich an das System zu gewöhnen. Dies ist eine Seite, welche die Änderungen zwar speichert und somit das Resultat sichtbar macht, aber in regelmäßigen Abständen geleert wird. Nahezu jedes Wiki stellt darüberhinaus eine Volltext- oder Titelsuche zum schellen Finden von Artikeln bereit.

#### 2.2.2.6 Semantische Wikis

In den letzten Jahren haben "Semantische Wikis" als Verknüpfung von Wiki-Konzepten mit semantischen Technologien zunehmend an Bedeutung gewonnen. Diese sollen an dieser Stelle kurz erwähnt werden. Im Allgemeinen vereinen Semantic Wikis Methoden des Semantic Web mit den Funktionalitäten eines normalen Wiki
[Schaffert et al., 2008]. Das Semantic Web stammt, wie auch das WWW, von Tim Berners-Lee und stellt eine Erweiterung des WWW dar, um die Bedeutung von Informationen für Maschinen bzw. Computer verwertbar zu machen
[Berners-Lee and Hendler, 2001]. Semantic Wikis bringen die Texte und Beziehungen unter den Seiten in ein maschinenlesbares Format, um sie für andere Services und Anwendungen nutzbar zu machen [Ebersbach et al., 2008a].

Die Flexibilität von herkömmlichen Wikis bei der Bearbeitung von Texten wird mit Semantic Wikis auf strukturierte Daten ausgeweitet. Dies wird durch die Unterstützung von Metadaten in Form von semantischen Annotationen (zu Wiki-Seiten als auch zu Verknüpfungen zwischen den Seiten) erreicht. Neben Texten können

auch Bilder, Audio-Dateien oder sonstige Multimedia-Inhalte annotiert werden. Diese Annotationen korrespondieren i.d.R. mit einer Ontologie [Schaffert et al., 2008]. Ontologien dienen der Wissensrepräsentation und bezeichnen eine Spezifikation einer Konzeptualiserung einer Domäne. Ontologien bestehen aus Konzepten und Relationen. Konzepte bezeichnen Typen von Entitäten und sind durch Relationen miteinander verbunden. Neue Relationen und Zusammenhänge aus der Ontologie lassen sich über Regeln ableiten und abfragen [Gams and Mitterdorfer, 2009]. In Semantic Wikis wird die Ontologie über die einzelnen Wiki-Seiten erstellt und gewartet. Dabei ist jedem Konzept eine Wiki-Seite zugeordnet. Die Konzepte werden über Links bzw. Annotationen verknüpft.

Intern werden die Annotationen in RDF[59]/OWL[60] repräsentiert, was den Austausch von Daten mit anderen Anwendungen erleichert. Die Ziele solcher Semantic Wikis sind unterschiedlich: Zum einen die Vereinfachung der kollaborativen Arbeit sowie der Navigation durch Annotationen ("Einsatz von Semantic Web-Methoden für Wikis"), zum anderen die kollaborative Erstellung von Ontologien ("Konzeption von Wikis für das Semantic Web") (vgl. [Krötzsch et al., 2007], [Schaffert et al., 2008]).

Nach [Schaffert et al., 2008] bieten Semantic Wikis nebst herkömmlichen Wiki-Funktionen folgendes (je nach Semantic Wiki Anbieter unterschiedlich stark implementiert):

- Einen einfachen Formalismus zur semantischen Annotation von Links und von Wiki-Artikeln.
- Eine semantische Suche, die neben Schlüsselwörtern auch nach semantisch zusammenhängenden Inhalten suchen kann (z.B. Oberbegriffe des Suchwortes).
- Eine Extraktion von Metadaten aus den Inhalten von HTML-Webseiten, die automatisch oder halbautomatisch erfolgt.
- Unter Umständen ein Verfahren zur automatischen Berechnung von Schlussfolgerungen aus den semantischen Annotationen [Schaffert et al., 2007].

Anhand eines Beispiels soll demonstriert werden, wie die Annotationen im Semantic MediaWiki[61] eingesetzt werden. Abbildung 2.2 zeigt einen Teil des Quelltexts des Wiki-Artikels "Denny Vrandecic" des Wikis SemanticWeb.org[62]. Exemplarisch sind

[59]Resource Description Framework ist eine Auszeichnungssprache für Metadaten, die in Form von Tripeln, bestehend aus Subjekt, Prädikat und Objekt abgelegt werden [Gams and Mitterdorfer, 2009]

[60]Web Ontology Language ist eine formale Beschreibungssprache zur Erstellung und Verteilung von Ontologien im WWW

[61]`http://semantic-mediawiki.org/`

[62]`http://semanticweb.org/wiki/Denny_Vrandecic`

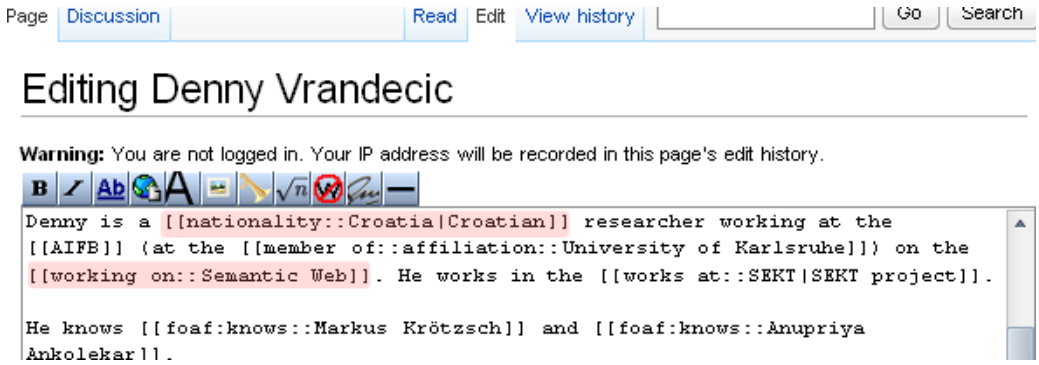



Abbildung 2.2: Annotationen am Beispiel des Semantic MediaWiki, Quelle: SemanticWeb.org

zwei Annotationen rot hinterlegt: Die Annotation `[[nationality::Croatia]]` kennzeichnet, dass zwischen den Konzepten "Denny Vrandecic" und "Croatia" eine Relation namens "nationality" existiert. Damit wird ausgedrückt, dass das Konzept "Denny Vrandecic" die Nationalität Kroatien hat, was darauf schließen lässt, dass es sich um eine Person handelt. Gleiches gilt für die Annotation `[[working on::Semantic Web]]`: Sie charakteristiert eine Relation zwischen den Konzepten "Denny Vrandecic" und "Semantic Web" mit dem Namen "working on", womit ausgedrückt werden kann, dass "Denny Vrandecic" an "Semantic Web" arbeitet. Diese Annotationen können nun z.B. in der semantischen Suche verwendet werden. Damit können Wiki-Inhalte anhand von Annotationen und Zusammenhängen gefunden werden.

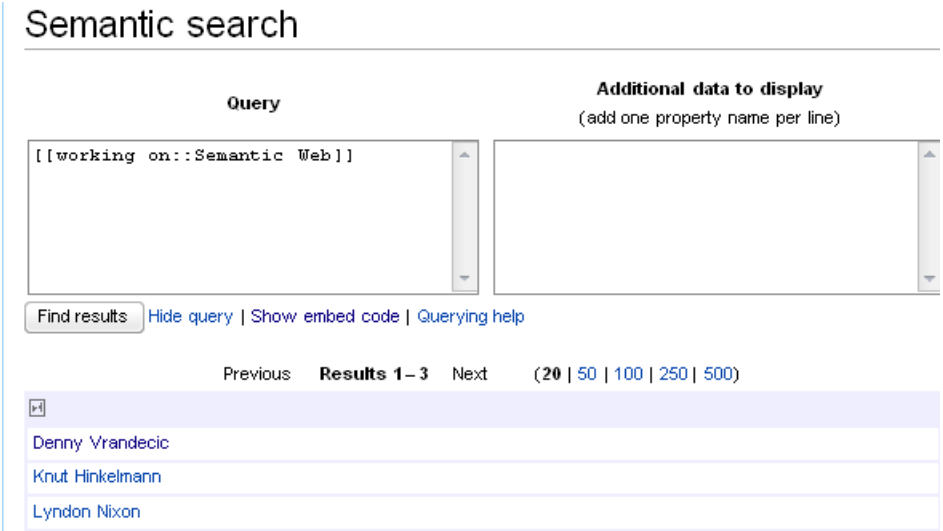



Abbildung 2.3: Semantische Suche des Semantic MediaWiki, Quelle: SemanticWeb.org

In Abbildung 2.3 ist die semantische Suche von SemanticWeb.org[63] zu sehen. Das Beispiel zeigt die semantische Suche nach allen Konzepten, die an "Semantic Web" arbeiten . Dazu wird als Suchanfrage (Query) die Annotation `[[working on::Semantic Web]]` eingegeben. Unten werden die Suchresultate dargestellt und drei Konzepte angezeigt. Darunter ist auch das Konzept "Denny Vrandecic" zu finden. Diese beiden Beispiele sollen zeigen, wie Semantic Wikis eingesetzt werden und Benutzer bei ihren Aufgaben unterstützen können. Die Möglichkeiten der Diskussion in und mit Wikis werden im folgenden Abschnitt behandelt.

#### 2.2.2.7 Potenzial der Diskussion i.A.

Arbeiten verschiedene Benutzer kollaborativ an einem Artikel bzw. einer Wiki Seite, so kommt es unweigerlich zu Unstimmigkeiten inhaltlicher Natur. Im Rahmen einer Diskussion können Missverständnisse geklärt sowie Unklarheiten beseitigt werden. Nahezu jedes Wiki bietet heute Möglichkeiten der Diskussion an. Wurden in frühen Wikis Diskussionen über den Inhalt als Teil der Wiki Seite geführt, so gibt es in neueren Wikis eigene Bereiche für die Diskussion [Farkas, 2007]. Dadurch ist nicht nur der Text leserlicher, auch die Kommunikation ist weniger ermüdend und zugleich konstruktiver. Zudem sind Leser eines Wiki Artikels nicht an der Diskussion um die Weiterentwicklung eines Themas interessiert, wenn sie sich nur zum Thema informieren möchten [Ebersbach et al., 2008a].

Je nach Wiki-Software wird Diskussion anders ermöglicht: Zum einen werden Diskussionen auf einer Extra Wiki Seite geführt, zum anderen erscheinen Diskussionsbeiträge in einem eigenen Fenster. Auch können sie, ähnlich wie bei Blogs, unterhalb der Wiki Seite als Diskussion angehängt werden [John et al., 2005]. Des weiteren unterstützen gewisse Wikis auch Diksussionthreads, sodass man direkt auf Diskussionsbeiträge anderer Nutzer antworten kann. Das MediaWiki z.B. verwendet eigene Diskussionsseiten. Diese sind eigene Wiki Seiten und den Artikelseiten ähnlich. Hier sind die Diskussionsbeiträge aller User auf einer Seite verpackt und in Wiki-Syntax geschrieben. D.h. jeder User kann die Kommentare anderer User editieren. Dies ist zwecks Nachvollziehbarkeit für andere User nicht sinnvoll und sollte deswegen vermieden werden. Es gehört zum guten Ton, entweder am Ende der Diskussionsseite einen neuen Beitrag hinzuzufügen oder seine Meinung direkt unter einen anderen Kommentar zu schreiben, wenn man diesen referenzieren möchte [Ebersbach et al., 2008a].

Wikipedia, welche die Wiki Engine MediaWiki einsetzt, hat Konventionen für die Diskussionseiten aufgestellt, um eine einheitliche, übersichtliche und effiziente Diskussion zu ermöglichen. Diese können auch auf andere Wikis angewendet werden, und einige sinnvolle Punkte daraus werden nachfolgend erwähnt. Um nachvollzie-

[63] Zu finden unter: `http://semanticweb.org/wiki/Special:Ask`. Aus Gründen der Übersichtlichkeit wurden unbedeutende Details in der Abbildung weggelassen.

hen zu können, von welchem User welcher Kommentar stammt, ist es sinnvoll seine Kommentare zu signieren. Entweder wird dazu der Benutzername, die IP-Adresse oder der echte Name eines Nutzers verwendet, wenn dieser kein Benutzerkonto besitzt. Um den Konversationsfluss deutlich zu machen, sollten Beiträge, welche sich auf einen anderen beziehen, eingerückt werden. In Wikipedia wird dies mit einem Doppelpunkt erzielt; je nach Wiki Engine wird dies jedoch aufgrund der unterschiedlichen Wiki Syntaxen anders gemacht, sofern die Engine solch eine Formatierung unterstützt. Verschiedene Diskussionsthemen sollten durch eine Überschrift getrennt werden. Dadurch entsteht eine Forum-ähnliche Struktur [LKD, 2009].

Die Diskussion kann durch unerfahrene Wiki Nutzer u.U. auch im Artikel selbst geführt werden, wodurch Artikel unleserlicher werden. Ein Wiki mit einer großen Community wird sich selbst regulieren und den User darauf aufmerksam machen, die Diskussionsseite anstatt der Artikelseite selbst zu verwenden. Allerdings könnte die Selbstregulation bei Wikis mit kleineren Communities und weniger engagierten Benutzern nicht so funktionieren und im Zusammenhang mit Diskussionsseiten bei neuen Benutzern immer wieder zu Problemen führen.

[Ebersbach et al., 2008b] haben im Rahmen einer Untersuchung über den Einsatz von Wikis im unternehmerischen Umfeld festgestellt, dass Kommentare wider Erwarten keine Einstiegsfunktion in die Wiki-Arbeit sind. In diesem Zusammenhang weisen die Autoren allerdings auf die räumliche Nähe der Beteiligten hin, durch die Unstimmigkeiten auf Zuruf und direkter Kommunikation einfacher und schneller behoben werden können. Kommentare in Wikis machen demnach bei asynchroner Arbeit mehr Sinn. [Kranz and Merz, 2005] haben Wikis in der Softwareentwicklung eingesetzt und die Erfahrung gemacht, dass Wikis als Dikskussionsforum nicht geeignet sind.

#### 2.2.2.8 Potenzial der Diskussion für den Anwendungsfall

Beim Einsatz eines Wiki Systems zur Diskssion von strukturierten Informationen im Anwendungsfall treten ähnliche Probleme wie bei Blogs auf. Diese betreffen zum einen die Modellierung bzw. Erstellung eines Erfolgsprofils und zum anderen die Diskussion selbst. Da es darum geht, Zusammenhänge zwischen einzelnen Erfolgsfaktoren zu diskutieren, muss als erster Schritt das Erfolgsprofil in einer Wiki Seite beschrieben werden. Natürlich ist zu beachten, dass nicht die gesamte Struktur bzw. kein vollständiges Erfolgsprofil aufgrund der Restriktion der Geheimhaltung offengelegt werden soll. Um dieser Anforderung gerecht zu werden, reicht es, nur einen Teilaspekt des Erfolgsprofils im Wiki zu schildern.

In diesem Zusammenhang wird die Kernfunktion eines Wikis zum Problem: Durch die Möglichkeit jedes Benutzers, vorhandene Wiki Seiten zu bearbeiten, kann die ursprüngliche Sicht eines Autors auf das Problem verfälscht bzw. anders dargestellt

werden, als dieser es ursprünglich vorgehabt hatte. Es muss bereits hier darauf geachtet werden, etwaige Änderungen auf der jeweiligen Diskussionsseite und nicht der Artikelseite selbst zu besprechen. Es ist auch das erklärte Ziel, eine Diskussion zu entfachen.

Aufgrund des Charakters von Wikis liegt es nahe, für Detailinformationen zu den Erfolgsfaktoren eigene Wiki Seiten zu verwenden. So können hier Informationen zum Charakter, eine Beschreibung sowie weiterführende Informationen und Links angegeben werden. Durch eine detaillierte Beschreibung werden Unklarheiten ausgemerzt, allerdings hat dies auch einen Nachteil. Innerhalb einer Wiki Seite können Verweise auf Seiten erzeugt werden, ohne dass diese bestehen. Klickt ein User auf so einen Verweis, so wird die Seite neu erstellt, und es ergibt sich noch kein Problem. Wird allerdings auf einen bestehenden Erfolgsfaktor und damit eine existierende Wiki Seite verwiesen, so muss diese Seite zwar nicht ein zweites Mal erstellt werden. Allerdings betreffen dann Änderungen des Erfolgsfaktors auch alle anderen Erfolgsprofile, die auf diesen verweisen und diese werden somit inkonsistent. Ein Beispiel soll dies verdeutlichen: User A deklariert den Erfolgsfaktor "Mitarbeitermotivation" auf einer eigenen Wiki Seite als Schwäche. User B erzeugt ein neues Erfolgsprofil, verweist auf die Seite "Mitarbeitermotivation" und sieht, dass diese bereits existiert. In seinem Erfolgsprofil ist der Erfolgsfaktor allerdings eine Stärke, und so bearbeitet er die Seite und ersetzt "Schwäche" durch "Stärke". Wenn also eigene Wiki Seiten für Erfolgsfaktoren verwendet werden, so sollten nur allgemeingültige Beschreibungen darin verfasst werden. Problem-spezifische Details wie Charakter (Stärke, Schwäche) müssen entweder im Erfolgsprofil selbst oder auf einer separaten Wiki Seite angemerkt werden, um Inkonsistenzen zu vermeiden.

Desweiteren spielt auch eine schwache Normierung in Wikis eine Rolle. Da Ausdrücke von Autor zu Autor verschieden geschrieben werden, kann es passieren, dass Seiten doppelt angelegt werden. Bestimmte Erfolgsfaktoren wie z.B. "Mitarbeitermotivation" treten mit hoher Wahrscheinlichkeit nicht nur einmal im gesamten Wiki auf. So kann es vorkommen, dass von einem Autor die Seite "Motivation der Mitarbeiter" angelegt wird, während schon eine andere Seite mit dem Titel "Mitarbeitermotivation" existiert. Weiters können bestimmte Begriffe abgekürzt werden, wie z.B. "WM" für "Wissensmangement". Aus diesem Grund ist es wichtig, sich auf ein gemeinsames Vokabular bzw. Regeln zu einigen. Viele Wiki Engines unterstützen sog. Weiterleitungen, womit z.B. im Falle eines Synonyms auf eine andere Wiki Seite weiterleiten kann [WikiMatrix, 2009b]. Für eine Weiterleitung reicht meist eine einfache Anweisung auf der Wiki Seite. Die "Podcast" Seite der Wikipedia[64] z.B. leitet so mit dem Wiki Quelltext $\#REDIRECT[[Podcasting]]$ auf die Seite "Podcasting"[65] weiter. Unter der Überschrift der Seite Podcasting ist zur Information ein Hinweis "Weitergeleitet von ..." angeführt. Mit diesen Redirects kann somit dem

[64] http://de.wikipedia.org/w/index.php?title=Podcast&redirect=no
[65] http://de.wikipedia.org/w/index.php?title=Podcasting

vorhin erwähnten Problem entgegengewirkt werden, wobei die Wiki Engine solche Weiterleitungen nicht automatisch durchführt. Die Benutzer müssen diese selbst erstellen, indem der Wiki Text der betreffenden Seite bearbeitet wird.

Wenn Diskussionsseiten statt einer sog. Threaded Diskussion unterhalb der Wiki Seite verwendet wird, so gilt das gleiche wie für herkömmliche Wiki Seiten: Kommentare können verändert und somit absichtlich falsche Informationen gepostet werden. Nach [Leuf and Cunningham, 2001] ist diese Freiheit aber eine Stärke und trägt positiv zur Kommunikation bei. Denn so könnnen Diskussionen überarbeitet werden, indem Kommentare neu angeordnet oder zusammengefasst werden, wenn es die Situation erlaubt. Weiters ist zu bedenken, dass zusätzlich die Wiki Syntax beherrscht werden muss. Doch nachdem Benutzer auf Diskussionsseiten mit wenigen wichtigen Formatierungen und Strukturierungen auskommen, stellt dies keine große Hürde dar.

Auch der Einsatz eines Semantic Wikis schafft keine Abhilfe. Artikel sind in Semantic Wikis zwar semantisch annotiert bzw. müssen diese oft erst durch die Benutzer annotiert werden. Dies ist eine zeitaufwändige und komplexe Aufgabe. Subjekte und Objekte sind eigene Seiten, womit sich das gleiche Problem bei der Erstellung des Erfolgsprofils wie bei herkömmlichen Wikis ergibt. Zudem ist die Usability nicht hoch, wenn Annotationen im Quelltext selbst hinzugefügt werden müssen. Nach [Ebersbach et al., 2008a] "hantieren Wiki-Normalverbraucher generell ungern mit Formalismen", womit der Akzeptanzgrad auf breiter Ebene nicht gegeben ist. Das IkeWiki[66] bietet zwar die Eingabe von Annotationen über Formulare an, aber die Eingabe ist dennoch nicht trivial und verlangt eine gewisse Kenntnis über die Annotationen.

### 2.2.3 Webforen

Dieser Abschnitt behandelt Diskussionsforen, die im WWW betrieben werden. Je nach Literatur werden verschiedene Ausdrücke für Diskussionsforen im Internet teilweise synonym verwendet. Gebräuchlich sind unter anderem Online Forum, Internetforum, Webforum, Bulletin Board, Message Board, Board, Newsgroup (vgl. [Pawlowitz, 2001], [Koch and Richter, 2008], [Shi et al., 2009]). In diesem Abschnitt wird versucht, Klarheit zu schaffen und die Unterschiede sowie Gemeinsamkeiten der verschiedenen Termini zu beschreiben.

#### 2.2.3.1 Definition

Nach [Phillips, 2007] sind Webforen:

> *"Web-based discussion boards are developed by participation. Emerging from an empty page, the board evolves as people submit questions or items*

[66] `http://ikewiki.salzburgresearch.at/`

> *for discussion. Readers are able to respond with answers or solutions, present opinions, or links to additional resource websites. As questions are answered and items discussed the board becomes a specific library of information."*

Eine kurze und prägnante Definition für Diskussionsforen finden [Koch and Richter, 2008], die die grundlegende Funktion eines Forums hervorheben:

> *"Hier können Diskussionsbeiträge (Postings) veröffentlicht werden, die von anderen Benutzern gelesen und beantwortet werden können. [...] Da die Funktionsweise von Foren an die eines schwarzen Bretts erinnert, wird sie oft mit einem solchen verglichen (daher auch das Synonym Bulletin Board)."*

Es wird zwar das Synonym Bulletin Board für Foren verwendet, allerdings gibt es Unterschiede struktureller Natur, auf die etwas später eingeganen wird.

#### 2.2.3.2 Geschichte

Ein Forum war der "ursprüngliche Ort der politischen Zusammenkunft im antiken Rom" [Schuegraf and Meier, 2005]. Heute können Diskussionsforen einerseits über das dezentral verwaltete Usenet realisiert werden, andererseits über das WWW als Webforen [Koch and Richter, 2008].

Elektronische Diskussionsforen existieren bereits seit Ende der Siebziger Jahre. Das Computerized Bulletin Board System (CBBS) wurde von Ward Christensen und Randy Suess 1978 entwickelt und war das erste elektronische Netzwerk zum Senden von Nachrichten. Es war ein textbasiertes System, mit dem Benutzer Nachrichten lesen und beantworten konnten. Über das Telnet Protokoll konnte man sich zu BBS-Servern verbinden und so mit anderen Benutzern Nachrichten austauschen [Moschovitis, 1999].

Nahezu zur gleichen Zeit wurde das Usenet entwickelt. Dieses Netzwerk stellt sog. Newsgroups bereit, über die man sich aus dem universitären und wissenschaftlichen Bereich Anfang der Achtziger austauschte (vgl. [Runkehl et al., 1998], [Pawlowitz, 2001], [Saade and Huang, 2009]). Das Usenet ist ein weltweites, elektronisches Netzwerk zum Austausch von Nachrichten über das NNTP-Protokoll[67], das aus einem dezentralen System von Servern besteht. Es ist unabhängig vom Internet, stellt sich jedoch für Newsgroup Nutzer als Teil dessen dar, weil das Internet als bevorzugter Transportweg für die Nachrichten dient. Jeder Nutzer kann sich daran beteiligen und das Prinzip des Dienstes ist einfach: Ein Nutzer sendet eine Nachricht an eine bestimmte Newsgroup, die andere Nutzer abrufen und gegebenfalls darauf antworten können. Dazu wird ein eigenes Programm, ein sog. Newsreader, verwendet. Die Newsgroups

[67] Network News Transfer Protocol

sind meist unmoderiert und kommen somit ohne die Kontrolle eines Moderators aus. Newsgroups gibt es nahezu zu jedem Thema. Aufgrund der großen Anzahl an Newsgroups werden die Themen nach Bereichen klassifiziert und durch Kategorien hierarchisiert. Damit ist ein besserer Überblick gewährleistet [Runkehl et al., 1998]. Doch mit der Entwicklung des World Wide Web verloren beide Systeme - vor allem das CBBS - an Bedeutung und Webforensysteme lösten diese als Diskussionsplattformen weitgehend ab [Moschovitis, 1999].

#### 2.2.3.3 Grundlagen

Webforen sind meist Teil einer Website oder eigenständige Webanwendungen. Im Gegensatz zu Newsgroups wird kein Newsreader für den Zugriff benötigt, sondern es reicht ein gewöhnlicher Browser. Nachrichten werden bei Webforen zentral an einer Stelle gespeichert und nicht wie beim Usenet über mehrere Server repliziert, wenngleich die Grundidee der des Usenet ähnelt [Koch and Richter, 2008]. Viele Foren setzen eine Registrierung für die Teilnahme voraus. Meist ist zwar das Lesen von Beiträgen in Foren auch unregistrierten Usern gestattet, das Verfassen von Postings allerdings nicht [Ebner, 2008].

Nach [Ebner, 2008] lassen sich vier Foren-Typen unterscheiden, was den Verwendungszweck betrifft:

- Die freie Community: Freie, nicht-kommerziell betriebene Internet-Communities bilden den Großteil der eingerichteten Foren im Web. Sie sind durch eine kostenlose Registrierung sowie keinerlei Benutzungsgebühren charakterisiert und werden meist von einer kleinen privaten Gruppe oder Einzelpersonen ins Leben gerufen. Eine durchwegs große Anzahl von Teilnehmern diskutiert entlang der thematischen Ausrichtung des Forums. Dabei kann sich jeder an der Diskussion beteiligen.

- Die abhängige Community: Eine abhängige Community wird von einer Körperschaft, wie einer Firma, Partei oder Rundfunkanstalt, einem Verlag oder ähnlichem betrieben, die für die inhaltliche Ausrichtung verwantwortlich zeichnet. Zum Teil zensieren die Betreiber die Inhalte, wenn diese Gegenstand der Kritik werden.

- Das Support-Forum: Um Kosten zu sparen, verwenden diverse Unternehmen ein Forum für den Kundensupport ihrer Produkte. Wenn eine Frage einmal öffentlich beantwortet wird, können sich auch andere Kunden dazu informieren. In diesen "Diskussionen" sind meist nur zwei Personen involviert: Der Kunde, der eine Frage stellt oder eine Anregung verfasst, und ein Mitarbeiter des Unternehmens, der darauf antwortet. In manchen Fällen bleibt der Forenbetreiber im Hintergrund, um eine Diskussion zwischen den Kunden zu entfachen damit diese sich selbst helfen. Denn dies spart ebenfalls Kosten.

- Interne Foren: Foren können auch für interne Arbeitsgruppen im unternehmerischen und privaten Umfeld eingesetzt werden. Es kann vor allem dann die Produktivität steigern, wenn die Mitglieder räumlich oder zeitlich getrennt arbeiten. Gegenüber einer Abstimmung per E-Mail haben Foren einige Vorteile. Da die gesamte Diskussion auf dem Server gespeichert ist, ist diese jederzeit nachvollziehbar. Bei E-Mail Kommunikation entstehen durch das häufige Antworten von E-Mails zum Teil sehr lange, unübersichtliche und schwer lesbare Diskussionen, die sich durch den Einsatz von Foren vermeiden lassen. Auch können sich neue Teilnehmer problemlos in die Diskussion einlesen und später daran teilnehmen. Vor allem für Vorgesetzte ist dies ein positiver Aspekt, da ihre Mailbox dadurch nicht mit unwichtigen E-Mails überfüllt wird. Sie entscheiden selbst, wann sie in die Diskussion einsteigen. Probleme mit Spam-Filtern oder nicht zustellbaren E-Mails bestehen auch nicht.

Nach [Koch and Richter, 2008] eignen sich Foren vor allem für Ankündigungen und wenn Anfragen in bestimmten Communities oder Themenbereichen gestellt werden sowie für ausschweifende Diskussionen.

Foren lassen sich nach der Strukturierung der Beiträge in zwei Typen einteilen: In klassische Foren und Bulletin Boards (kurz Boards). Während in klassischen Foren die Beziehungen zwischen Postings innerhalb eines Themas in einer Baum-Struktur dargestellt werden, vereint ein Board alle Beiträge eines Themas auf einer Seite (vgl. [Münz, 2007], [Koch and Richter, 2008]).

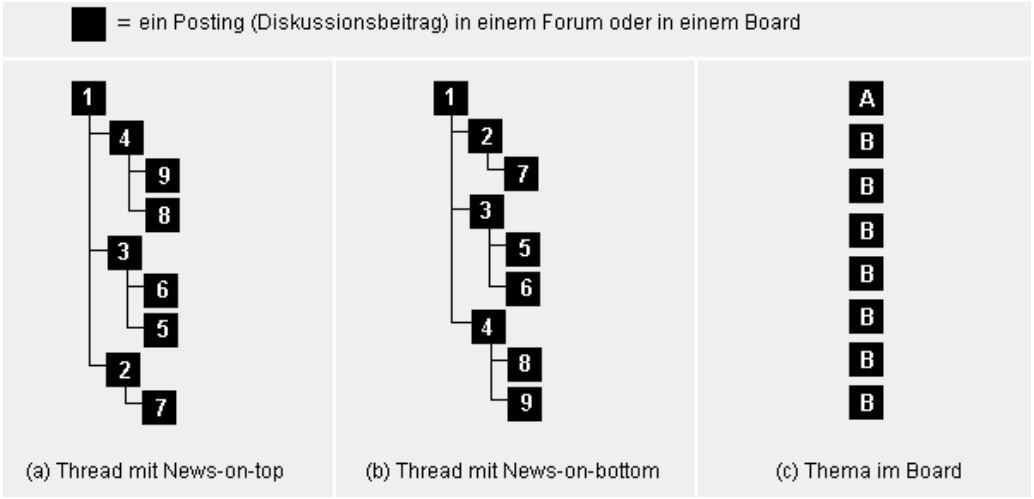



Abbildung 2.4: Gegenüberstellung der Struktur von Threads und Board Postings: (a) ein Thread in einem klassischen Webforum als Baumstruktur mit zeitlicher News-on-top-Einordnung; (b) ein Thread mit zeitlicher News-on-bottom Einordnung; (c) ein Thema und Beiträge dazu in einem Board (entnommen aus [Münz, 2007])

In Abbildung 2.4 werden die unterschiedlichen Strukturen von Thread und Board-

Beiträgen gegenübergestellt. (a) und (b) zeigen je einen Thread (zu deutsch: Faden) eines klassischen Forums. Dieser Diskussionsfaden ist die gedankliche Basis-Einheit eines Forums und beschreibt, wie die Beiträge miteinander verbunden sind. Er dient der Visualisierung der Diskussion und wird als Baumstruktur dargestellt. Dies hilft vor allem neuen Teilnehmern, sich ein Bild von der Diskussion zu machen. In der Abbildung sind die schwarzen Quadrate Postings zu einem Thema, die Verbindungen bezeichnen direkte Antworten auf den vorhergehenden Beitrag. Für eine optimale Darstellung spielt auch die zeitliche Reihenfolge eine Rolle, welche durch die Zahlen in den Postings repräsentiert wird. Ganz oben an der Wurzel des Baumes in Abbildung 2.4 (a) und (b) ist das Ausgangsposting mit der Zahl 1 und damit das älteste. Bei der Darstellung von zeitlichen Informationen hat sich das Prinzip "newest on top" etabliert, was bedeutet, dass jüngere Informationen immer oben stehen. Bei Threads ist dieses "on top" relativ zu sehen, da die Struktur der Antworten mit ihren Verästelungen erhalten bleiben soll. Deswegen wird in (a) auch das Posting 4 direkt unter dem Ausgangsposting 1 dargestellt. Es ist zwar nicht das jüngste Posting innerhalb des Threads überhaupt, aber die neueste direkte Antwort auf den Ausgangsbeitrag. Dies gilt auch für alle weiteren Antworten auf den unteren Ebenen des Baumes. Auf den Beitrag 4 antworteten die Postings 8 und 9, wobei 9 später verfasst wurde, damit jünger ist und auch über dem Posting 8 steht. Auch wenn das Prinzip "newest on top" vorherrschend ist, so finden sich auch viele Foren mit einer herkömmlichen chronologischen Anordnung der Beiträge, sozusagen "newest on bottom". Neuere Antworten auf Beiträge werden dabei nicht ober- sondern unterhalb eines Beitrages gesetzt, wie dies in Abbildung 2.4 (b) zu sehen ist. In Foren existiert i.a.R. eine Übersicht aller Threads. Wird ein Thread von einem User ausgewählt, so kann dieser das Ausgangsposting lesen, während die Antworten darunter angezeigt werden. Die Überschrift der Antwortenwerden darunter angezeigt, um die Diskussionsstruktur besser zu erkennen. Antworten werden dabei eingerückt, was der Baumstruktur genügt. Wird eine Antwort ausgewählt, so wird wieder nur der Text dieser auf der Seite dargestellt, vom Ausgangsposting ist dann auch nur noch die Überschrift zu lesen. Die kleinste, als separate Webseite abrufbare Einheit ist somit ein einzelner Beitrag. [Münz, 2007].

Boards sind im Gegensatz zu Foren themenorientiert. Die gedankliche Basis-Einheit ist somit das Thema und nicht der Thread. Diese wird in Form einer Liste mit einer flachen Beitrasstruktur wie in Abbildung 2.4 (c) dargestellt. Der Beitrag A gibt dabei das Thema vor, alle B's sind weitere Beiträge, die den Charakter eines Statements haben [Münz, 2007]. Alle Beiträge eines Themas werden auf einer Seite vereint. Das Thema wird nach einer einstellbaren Anzahl von Beiträgen auf eine neue Seite umgebrochen [Koch and Richter, 2008]. Im Unterschied zu einem Forum ist aufgrund der Listendarstellung nicht zu erkennen, ob z.B. der fünfte Beitrag auf den ersten oder den vierten antwortet. Dazu muss der User den Inhalt des Beitrags lesen. Nachdem die Beiträge nicht eingerückt werden, sind diese links ausgerichtet, was dem gewohnten Lesebedürfnis entegegenkommt. Die Liste eines Themas ist ei-

ne fest zusammengehörige Einheit und auch die kleinste als eigenständige Webseite aufrufbare Einheit [Münz, 2007]. Augrund der hohen Anzahl und des großen Themenspektrums von Boards gliedern Board-Systeme die Beiträge hierarchisch. Die kleinste Einheit ist das Thema, welches einem Brett bzw. "Board" zugehörig ist. Ein Brett ist einem "Schwarzen Brett" nachempfunden, an das jeder seine Zettel bzw. Themen heften kann. Bretter werden ihrerseits in Kategorien zusammengefasst. So können Board-Teilnehmer durch Kategorien über Boards zu den Themen mit den Beiträgen navigieren [Ebner, 2008]. Nichtsdestotrotz haben sich klassische Foren nach [Koch and Richter, 2008] aufgrund der besseren Übersichtlichkeit der Beiträge gegenüber Boards durchgesetzt. Zwar ist die Abrufzeit eines Themas in einem Board i.d.R. niedriger, aber es ist schwieriger zu erkennen, wer auf welchen Beitrag geantwortet hat. Dies wird vor allem bei langen, komplexen Themen mit verschiedenen (Teil-)Diskussionen zum Nachteil.

#### 2.2.3.4 Technik und typische Funktionen

Forensysteme sind Anwendungen, die auf einem Webserver betrieben werden. Die Beiträge werden in einer Datenbank gespeichert, im weiteren Sinne sind sie Content Management Systeme [Ebner, 2008]. Wie Wikis werden sie in verschiedenen Programmiersprachen angeboten. Der Großteil der Foren Software-Lösungen sind in PHP implementiert, auch in Java und Perl sind einige Systeme verfügbar.
[ForumMatrix, 2009] listet zur Zeit mehr als 50 verschiedene Forensysteme auf, wovon die überwiegende Mehrheit als Open Source Software frei verfügbar ist. Die bekannteste und meist eingesetzte Open Source Forum Software ist phpBB[68] [phpBB, 2009]. Nahezu alle auf [ForumMatrix, 2009] aufgelisteten Forensysteme unterstützen BBCode (Bulletin Board Code) zur Textformatierung in Postings. BBCode ist wie Wiki Syntax eine Pseudo-Auszeichnungssprache, das von einem Forensystem eingeführt wurde und sich bewährt hat. BBCode beinhaltet eine Menge von Formatierungsregeln. Viele Forensysteme bieten Eingabehilfen beim Schreiben von Beiträgen an, wie man es von Textverarbeitungsprogrammen gewohnt ist [Münz, 2009].

Foren haben i.A. eine definierte Benutzerhierarchie von Administratoren, Moderatoren und Benutzergruppen. Administratoren verwalten das System, können neue Boards und Kategorien sowie Benutzergruppen anlegen und auch weitere Funktionen freischalten. Sie haben auf alle Bereiche eines Forums Zugriff, außer auf private Nachrichten. Administratoren weisen Benutzern verschiedene Benutzergruppen zu. Diese haben unterschiedliche Zugriffsrechte auf Boards. Wie und ob unterschiedliche Zugriffsrechte in einem Forum eingesetzt werden, ist jedem Betreiber selbst überlassen. Moderatoren werden zum Moderieren von Boards eingesetzt. Viele Foren unterstützen auch das Schreiben von privaten Nachrichten. Dies ist vergleichbar mit E-Mail, nur bleiben die Nachrichten im Kontext eines Forums und sind nur für den

[68] `http://www.phpbb.com/`

Empfänger sichtbar [Ebner, 2008].

Heutige Foren bieten weit mehr Funktionen an als frühere Forensysteme: Verbesserte Suchfunktionen wie Volltext-, Autoren- oder Kategoriensuche sind in vielen Foren mittlerweile üblich. Neben vielseiteigen Eingabehilfen zur Erstellung von Beiträgen wie Farbwahl sowie Schriftschnitt und -art, sind auch einige Systeme mit einer Rechtschreibprüfung ausgestattet [Harman and Koohang, 2005]. Neben herkömmlichem Text können vielmals auch Dateien an einen Beitrag angehängt werden. Hier sind allerdings meist aus Sicherheitsgründen nur wenige verschiedene Dateiformate wie Text- oder Bildformate erlaubt [ForumMatrix, 2009]. Benutzer können für sie interessante Threads abonnieren. So können sie am Laufenden gehalten und per E-Mail benachrichtigt werden, wenn auf einen Beitrag in einem abonnierten Thread geantwortet wird. Viele Foren unterstützen mittlerweile auch RSS-Feeds zur Benachrichtigung [Koch and Richter, 2008]. Statistiken geben Aufschluss darüber, wie stark ein Forum besucht oder wie viele Nachrichten in einem Forum oder einem Unter-Forum gepostet wurden. Benutzer werden unmittelbar informiert und sehen bereits in der Board-Übersicht, wie oft auf einen Beitrag geantwortet wurde oder welcher Beitrag der neueste ist [ForumMatrix, 2009].

#### 2.2.3.5 Potenzial der Diskussion i.A.

Wie auch bei Blogs und Wikis verläuft die Kommunikation in Foren asynchron, d.h. Produktion und Rezeption der Diskussionsbeiträge finden i.d.R. zeitversetzt statt. Der gesamte Kommunikationsablauf ist transparent und archiviert, der Zugriff auf die Inhalte auch nach Jahren noch möglich. Je nach Art des Forums sind diese unterschiedlich stark strukturiert (vgl. lineare Listen in Boards und Foren mit Baumstruktur) [Stockmann, 2004]. Auch wenn Foren oft aus "one-to-many-conservations" bestehen, sind auch hier Phasen von "one-to-one-conservations" zu finden. Die Beiträge in Foren sind zwar medial verbreitet, richten sich jedoch nicht wie massenmediale Inhalte an ein disperses Publikum. Der Adressatenkreis ist aufgrund der Ausrichtung auf die Beantwortung von Fragen oder das Abgeben von Statements zu einem Thema enger [Schuegraf and Meier, 2005].

Während in Wikis oder Blogs ein Artikel im Zentrum der Betrachtung ist, so steht in Foren und Boards die Diskussion selbst im Mittelpunkt. Ein Forum ist i.A. ein virtueller Ort, an dem es zu einem Austausch von Ideen und Gedanken kommt [Saade and Huang, 2009]. Hier initiiert ein Benutzer ein Thema oder stellt eine Frage und will so eine Diskussion anstoßen. Diskussionsbeiträge in Foren sind oft länger und der Diskussionsstrang selbst zum Teil übersichtlicher als auf Blogs und Wikis, da i.d.R. mehr Platz für die Posts auf der Webseite zur Verfügung steht. In vielen Foren können Beiträge nicht nur Text enthalten, es können auch Dateien wie z.B. Bilder angehängt werden. Somit erhält ein Statement oder Beitrag zusätzlichen Informationsgehalt und die Diskussion gewinnt an Qualität, welche bei einer Dis-

kussion auf einem Blog oder Wiki nicht erreicht werden kann. Die Beiträge werden an einer Stelle zentral gespeichert. Um auf dem neuesten Stand zu bleiben, muss das Forum nicht zwingendermaßen besucht werden: Benachrichtigungen per E-Mail sind mittlerweile Standard und immer mehr Foren bieten auch Feeds an. Eine automatische Benachrichtigung per Trackback bei der Referenzierung von Beiträgen wie bei Blogs besteht nicht. So ist die Vernetzung unter Foren schwach, allerdings wird dadurch die Diskussion nicht in ein anderes Umfeld verschoben.

Foren haben im Gegensatz zu Blogs einen Portal-Charakter [Harman and Koohang, 2005]. Forenmitglieder sind abgesehen von Administratoren und Moderatoren alle gleichberechtigt. Auf einem Blog schreibt i.d.R. nur ein Benutzer ein Post zu einem Thema seiner Wahl, die anderen User geben dazu lediglich Kommentare ab. Grundsätzlich besteht zwar die Möglichkeit, verschiedene Benutzer auf einem Blog einzurichten und Atrikel von verschiedenen Autoren verfassen zu lassen, wie dies z.B. von den News-Seiten Ajaxian[69] oder Mashable[70] praktiziert wird. Allerdings sind Ein-Autoren-Blogs die Mehrheit dar. In einem Forum hingegen kann jeder ein Thema initiieren. Die Themen sind somit an den Bedürfnissen und Interessen aller Benutzer orientiert.

#### 2.2.3.6 Potenzial der Diskussion für den Anwendungsfall

Die Diskussion von strukturierten Informationen im vorliegenden Anwendungsfall mit Foren wird im folgenden Szenario näher beschrieben. Foren können dabei ähnlich eingesetzt werden wie Blogs und Wikis, wobei vergleichbare Probleme autreten.

Zuerst muss ein Beitrag verfasst werden, in dem ein Erfolgsprofil beschrieben wird. Dieses bzw. ein Teilproblem davon wird dabei in textueller Form inklusvie Erfolgsfaktoren und Verbindungen skizziert. Wurde ein neues Thema erfolgreich im Forum veröffentlicht, so kann die Diskussion beginnen. Interessierte Benutzer können an der Diskussion teilnehmen, indem sie auf den Initialbeitrag antworten oder auf bereits abgegebene Beiträge anderer User eingehen. In klassischen Foren mit Thread-Darstellung können sich User einzelne Unterdiskussionen besser verfolgen: Hier ist leicht zu erkennen, welche Antwort sich auf welchen Beitrag bezieht. Bei Verwendung eines Bulletin Boards werden zur thematischen Gliederung und besseren Auffindbarkeit von Beiträgen Boards eingerichtet.

Da viele Foren über eine integrierte Benutzerverwaltung inklusive Möglichkeiten der Zugriffsbeschränkung verfügen, empfiehlt es sich, das Schreiben von Nachrichten nur registrierten Nutzern zu gestatten. Dies stellt zum einen eine Barriere für anonyme Benutzer dar, Spam Nachrichten zu verfassen. Diese müssen sich zumindest registrieren und anmelden, bevor sie Beiträge verfassen dürfen. Bei Bekanntwerden ei-

[69] `http://ajaxian.com/`
[70] `http://mashable.com/`

ner Spam-Identität kann diese gesperrt oder gelöscht werden. Zum anderen können so Statistiken z.B. über die Aktivität eines jeden Mitglieds erstellt werden. Zum Dritten trägt es positiv zur Communitybildung bei, da Benutzer im Lauf der Zeit die anderen Mitglieder besser kennen lernen, da sie sehen, welche Meinungen diese vertreten. Auch die Kontaktaufnahme zu einem Mitglied wird dadurch erleichtert. Wenn eine Diskussion nur auf bestimmte Benutzer beschränkt werden soll, können auch private Boards eingerichtet werden.

Nachdem in einem Forum jeder (registrierte) Benutzer Nachrichten verfassen darf, ist die Wahrscheinlichkeit, dass ein Thema oder ein Teilaspekt davon schon in einem bestehenden Thread behandelt wurde, größer als bei einem Blog, den nur ein Autor betreibt. Natürlich ist dies abhängig von Faktoren wie Reichweite bzw. Besucher der Seite sowie Aktivität und Bereitschaft der Nutzer, Beiträge zu liefern. Dennoch können User in einem Forum initiativ werden und selbst ein gewünschtes Thema starten. Hilfreich beim Finden bzw. Suchen von ähnlichen Problemen ist die in vielen Foren-Systemen vorhandene Volltextsuche. In einem Discussion Board können Nachrichten in den kategorisierten Boards auch direkt durchstöbert werden.

### 2.2.4 Fazit

Die drei vorgestellten Plattformen Blogs, Wikis und Foren erlauben zwar die Diskussion über Kommentare bzw. Beiträge in textueller Form, haben allesamt jedoch das gleiche Problem: Bestehende strukturierte Informationen müssen von Benutzern in textuelle Form überführt werden, um eine Diskussion zu ermöglichen und so Anregungen von Dritten zu erhalten. Der Mehrwert der Struktur geht in diesem Fall verloren. Vor allem aber sind das Feedback bzw. die Beiträge und Kommentare der Benutzer unstrukturiert. Sollen diese Informationen genutzt oder weitverarbeitet werden, so müssen wichtige Details daraus entweder durch Benutzer selbst extrahiert oder mittels Text Mining Techniken analysiert und extrahiert werden. Ersteres ist zeitaufwendig und ab einer gewissen Größenordnung von Menschen nicht mehr bewältigbar. Abgesehen von einem erheblichen Aufwand, Text Mining zu implementieren, ist vor allem die erreichte Genauigkeit durch Text Mining gering. Text Mining ist eine Heuristik und nur im statistischen Mittel aussagekräftig. Schon alleine die Extraktion von Zahlen, den dazugehörigen Einheiten und was sie aussagen ist schwierig. Das folgende Beispiel mit vier verschiedenen Sätzen, die alle die gleiche Bedeutung haben, demonstriert dies: “Die Kosten dafür belaufen sich auf 12k €”, “Kostenpunkt: 12.000 €”, “Cost factor: 12,000 €”, “Der Aufwand beläuft sich auf zwölftausend Euro”. Mit automatischen Verfahren ist die Bedeutung schwer zu extrahieren und noch dazu fehlerbehaftet. Für akkurate Analysen benötigt man strukturierte Information. Aus diesen Gründen muss eine andere Möglichkeit zur Diskussion der vorliegenden strukturierten Informationen gefunden werden.

# Kapitel 3

# Rich Internet Application als Umsetzungsplattform

Die bisher erwähnten Plattformen sind für den Anwendungsfall nicht einsetzbar. Es muss eine Plattform geschaffen werden, die die bestehenden, strukturierten Informationen aufbereitet und dem Benutzer so präsentiert, dass eine Diskussion in strukturierter Manier möglich ist. Dies könnte durch Bewertungen oder auch Tags realisiert werden, zusätzlich könnten Kommentare in Textform angeboten werden. Zudem muss so eine Plattform ein sehr hohes Maß an Usability aufweisen, um Benutzer zu strukturierten Eingaben zu motivieren. Rich Internet Applications sind eine Art von Anwendungen, die diesen Anforderungen gerecht werden. Diese werden nachfolgend näher betrachtet. Danach wird das Konzept erklärt, das die Anforderungen aus dem ersten Kapitel noch einmal zusammenfasst. Danach folgt die Beschreibung, wie diese umgesetzt werden.

## 3.1 Rich Internet Applications

### 3.1.1 Defintion

Der Ausdruck "Rich Internet Application" (kurz RIA) wurde das erste Mal von Jeremy Allaire in 2002 genannt [Allaire, 2002]. Der ehemalige Macromedia Mitarbeiter[1] beschreibt in dem White Paper Rich Internet Applikationen und vor allem technische Aspekte der Macromedia Flash MX Client Umgebung. Nach [Allaire, 2002] haben Rich Internet Anwendungen einen ähnlich hohen Unterhaltungswert wie Desktopanwendungen, sind jedoch gleichzeitig so kostengünstig und flexibel in der Bereitstellung wie herkömmliche Webanwendungen, da sie ebenfalls auf die plattformunabhängige Natur des WWW bauen (vgl. Abbildung 3.1). Auch wenn der Autor seine Ausführungen sehr auf die Flash Plattform bezieht, sind sie für sämtliche

[1]Das Unternehmen Macromedia entwickelte den Flash Player zum Betrachten von Flash Filmen bzw. Programmen und wurde 2005 vom Unternehmen Adobe übernommen [Krempl, 2005]. Der Macromedia Flash Player heißt seitdem Adobe Flash Player.

RIA-Plattformen gültig.

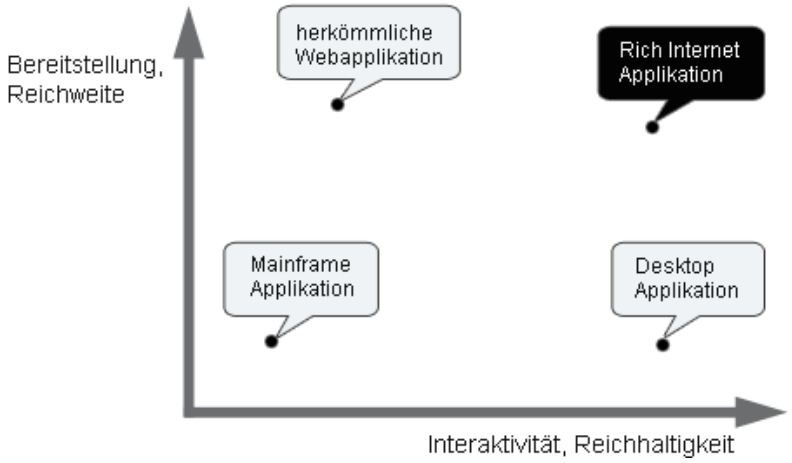



Abbildung 3.1: Reichweite (Bereitstellung) und Reichhaltigkeit (Interaktivität) von Rich Internet Applikationen (vgl. [Simmons, 2007])

Auch [Giametta, 2009] hat eine an Allaire angelehnte Definition von RIA:

> *"RIAs are a cross between traditional desktop applications and Internet-based applications. They blend together the best attributes of both application types [...] to provide richer user experiences (including audio, video, and communications), online and offline support, and more responsive applications. And, unlike standard web pages, RIAs do not require page reloading."*

Rich Internet Applications vereinen also die Vorteile von Desktop Anwendungen und herkömmlichen webbasierten Anwendungen. Das "Rich" bezieht sich dabei auf die "reichhaltigen" Bedienmöglichkeiten, die diese Benutzerschnittstellen bieten. Diese Reichhaltigkeit manifestiert sich in intuitiven, reaktionsschnellen und überzeugenden Benutzererlebnissen. Einen Grund technischer Natur für die Ermöglichung von solch reichhaltigen Benutzerschnittstellen nennt [Eichorn, 2006]:

> *"A Rich Internet Application [...] contains more code on the browser, which offers higher levels of interactivity and an experience similar to native applications."*

RIAs lagern einen erheblichen Teil des Codes auf den Client aus: Dies ist die sog. Client Engine, welche einen Teil der Anwendungslogik beinhaltet. Die Client Engine verarbeitet Benutzereingaben lokal und muss nur auf den Server zugreifen, wenn die Client Engine die Eingaben nicht verarbeiten kann. Dies geschieht viel seltener als bei herkömmlichen Webanwendungen und ist der Grund für die Reaktionsschnelligkeit von RIAs.

RIAs werden häufig auch unter folgenden Bezeichnungen genannt: Remote scripting, X Internet, Rich Web Clients und Rich Web Applications (vgl. [Giametta, 2009], [Allaire, 2002]).

### 3.1.2 Geschichte der RIAs

Zu Beginn war das WWW für den dokumentenbasierten Informationsaustausch konzipiert. Unternehmen erkannten das Potenzial des Webs schnell: Zu geringern Kosten und einer nie dagewesenen Reichweite konnten sie sich mit einem Webauftritt aller Welt präsentieren. Aufgrund der Gegenüberstellung von Kosten und Nutzen sowie der Vorteile der Plattformunabhängigkeit wurde das Web nach und nach als Plattform für Anwendungen genutzt. Die Webanwendungen erster Stunde basierten auf einem Client-Server-Modell. Eingesetzt wurden existierende Technologien wie das Übertragungsprotokoll HTTP[2] sowie die Seitenbeschreibungssprache HTML[3] zur Anzeige der Inhalte. Das Web entwickelte sich rasant, die Anforderungen der User an die Benutzerschnittstellen stiegen, herkömmliche Webanwendungen stießen jedoch an ihre Grenzen: HTML war grundsätzlich nicht für komplexe User Interfaces konzipiert. Auch wenn die Technologien (HTML, CSS) weiterentwickelt wurden, standen Webanwendungen Desktopapplikationen in Sachen Usability und Interaktivität noch um vieles nach. Vor allem das vorherrschende Anwendungsmodell nach dem Request-Response-Prinzip basierend auf Webseiten war ein Problem: Nahezu jede Benutzereingabe erfordert eine Anfrage an den Server, der die Eingabe verarbeitet und als Resultat eine vollständige Webseite zurückliefert, die im Browser neu aufgebaut und dargestellt werden muss. Dadurch müssen nicht nur redundate Daten an den Client gesendet werden; der Benutzer muss während jeder Server-Anfrage warten, kann nicht weiterarbeiten und wird somit aus seinem Arbeitsfluss herausgerissen ( vgl. [Farrell and Nezlek, 2007], [Ehrenstein, 2009]).

Der RIA-Pionier Jeremy Allaire erkannte, dass die vorherrschenden Paradigmen der Webentwicklung keinen Mehrwert für die Benutzer mehr brachten und ihnen mit neuen Ansätzen Rechnung getragen werden musste. In [Allaire, 2002] nennt er erstmals den Begriff Rich Internet Application. Er definiert die Anforderungen an eine RIA und stellt die Flash MX Plattform vor, welche alle angeführten Anforderungen abdecken sollte. Heute weiß man, dass sich der Autor zwar geirrt hat, sich die Flash Plattform seither aber sehr eng an diesen Vorgaben weiterentwickelt hat[Ehrenstein, 2009]. Die Nachfrage nach RIAs ist seit deren Einführung stark gestiegen, heute sind sie kaum mehr aus dem WWW wegzudenken (vgl. [Simmons, 2007], [Giametta, 2009], [Ehrenstein, 2009]) und mit ein Grund für den Erfolg von Web 2.0 (vgl. Abschnitt 2.1.1.2). Neben den bereits mehrfach erwähnten reaktionsschnelleren und intuitiveren Benutzerschnittstellen nennt [Noda and Helwig, 2005] vier weitere

[2]Hypertext Transfer Protocol
[3]Hypertext Markup Language

gewichtige Gründe für den Erfolg von RIAs:

- Breitband-Internet: RIAs lagern einen nicht geringen Code-Teil auf den Client aus, der zu Beginn der Arbeit mit einer RIA heruntergeladen wird. Erst als die breite Masse an Internet Benutzern über schnelle Breitbandanschlüsse verfügte und die langsamen Modems ablösten, war dies keine Barriere mehr und auch dere Konsum von weiteren datenintensiven Diensten wie Audio- und Video-Streams wurde nahezu allen Nutzern ermöglicht.

- Prozessorleistung: Der Unterschied der Prozessorleistung zwischen PCs und Servern ist heute viel geringer als vor einigen Jahren noch. Nachdem RIAs mehr Arbeit auf der Clienseite erledigen als herkömmliche Webapplikationen, schaffen leistungsfähigere Client Computer ideale Voraussetzungen für RIAs.

- Plattformen: RIAs fristen kein Nischendasein. Viele Software Unternehmen haben die Vorteile und das Potenzial von RIAs erkannt und bieten Entwicklungs- und Lauftzeitplattformen an. Google setzt auf die AJAX-Technik bei seinen Applikationen Maps, Docs oder Mail[4]. Adobe bietet mit Flex[5] eine RIA-Entwicklungsumgebung für die Flash Plattform und mit AIR[6] eine plattformunabhängige RIA-Laufzeitumgebung für den Desktop an. Microsoft Silverlight[7] und JavaFX[8] von Sun sind zwei weitere RIA Plattformen von namhaften Software Herstellern. Mit OpenLaszlo[9] und Mozilla XULRunner[10] werden auch Open Source RIA-Plattformen angeboten.

- Web Services und SOA[11]: Das Aufkommen von Web Services und SOA hat die Entwicklung von Geschäftsanwendungen beeinflusst. Dadurch lassen sich Service- und Präsentationsschicht einer Anwendung entkoppeln. Nahezu jede RIA-Plattform bietet eine Präsentationsschicht, die entkoppelt vom Service Layer funktioniert und Module zur Kommunikation mit Web Services und Schaffung einer SOA ermöglicht.

### 3.1.3 RIA-Charakteristiken

[Allaire, 2002] definierte zwar bereits im Jahr 2002 die Anforderungen, welche an RIAs gestellt werden. Diese haben sich allerdings kaum geändert und können als die charakteristischen Eigenschaften von RIAs betrachtet werden. Im Folgenden werden diese beschrieben:

---

[4] `http://maps.google.com/`, `http://docs.google.com/`, `http://mail.google.com/`
[5] `http://www.adobe.com/de/products/flex/`
[6] Adobe Integrated Runtime, `http://www.adobe.com/products/air/`
[7] `http://www.microsoft.com/silverlight/`
[8] `http://www.javafx.com/`
[9] `http://openlaszlo.org/`
[10] XML User Interface Language Runner, `https://developer.mozilla.org/en/XULRunner`
[11] Serviceorientierte Architektur

- Performante Laufzeitumgebung: RIAs benötigen eine effiziente, hochperformante Laufzeitumgebung zur Ausführung von Code, der Anzeige von (multimedialen) Inhalten (Audio, Video) oder auch zur Ermöglichung von verschiedenartiger Kommunikation (Audio- oder Video-Chat). [Allaire, 2002] kreidete vor allem die Probleme von HTML zum damaligen Zeitpunkt an und präsentierte Flash MX als eine Lösung. Auf AJAX (und damit auch auf HTML) basierende RIAs konnten erst zwei bis drei Jahre später entwickelt werden, da das XMLHttpRequest-Objekt[12] erst Ende 2004 von Browsern auf breiter Basis unterstützt wurde [Schutta and Asleson, 2006]. Browser und deren JavaScript-Engines waren zudem nicht so performant (vgl. [Kane, 2009]). In den letzten Jahren legten die Browser Hersteller vermehrt Wert auf die Leistungsoptimierung der JavaScript-Engines (vgl. z.B. [Lindström, 2009], [Mozilla, 2009a], [ieblog, 2009]), was die Bedeutung einer performanten Laufzeitumgebung für AJAX-basierte und JavaScript-intensive RIAs und unterstreicht.

- Integration in eine Umgebung: Mit RIAs ist eine Integration von Inhalt, Kommunikation und Anwendungsschnittstellen in eine gemeinsame Umgebung möglich. RIA-Plattformen wie Silverlight oder Flash ermöglichen bereits heute die Einbettung von Multimedia Inhalten wie Video oder Audio. HTML Anwendungen lösen dies heute meist über das Flash Plugin (z.B: YouTube[13]). Doch die W3C-Spezifikation von HTML5, die gerade in Arbeit ist, sieht eine direkte Einbettung in HTML-Seiten und Applikationen von Video- und Audioinhalten samt Steuerungsfunktionen vor [Hickson and Hyatt, 2009]. So können in Zukunft[14] AJAX-basierte RIAs auf das Flash-Plugin zur Video-Einbettung unter Umständen verzichten.

- Objektmodelle: Eine RIA-Plattform muss leistungsfähige und erweiterbare Objektmodelle für Interaktivität zur Verfügung stellen. Serverseitige Plattformen wie Java oder .NET existieren schon seit langem und verfügen über performante Objektmodelle. Auf Client-Seite gibt es z.B. mit Adobe Flex ein effizientes und performantes Framework, das auf der Flash Plattform läuft. Die stetige Weiterentwicklung von JavaScript und des W3C Document Object Models (DOM) haben auch zu Performance Verbesserungen von AJAX-RIAs geführt.

- Rapid Application Development (RAD): Die Verwendung von Komponenten und Wiederverwertbarkeit von Code soll den Einsatz schlanker und skalierbarer Programmiermodelle wie RAD, Agile Softwareentwicklung oder Scrum

---

[12]Das XMLHttpRequest-Objekt ermöglicht asynchrone Anfragen an einen Weberver und in weiterer Folge das Laden von Daten in eine bestehende HTML-Seite mittels JavaScript.

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

[14]Wenn die btreffenden Spezifikationen in HTML5 komplett sind und dies von den wichtigen, verbreiteten Browsern unterstützt wird.

ermöglichen und forcieren. Der Entwicklungsprozess soll damit flexibler und schneller werden, da Anforderungen schneller umgesetzt werden können.

- Web Services: Nahezu jeder größere Anbieter von Webangeboten bietet heute APIs in Form von Webservices für den Zugriff auf dessen Daten an (vgl. 2.1.1.2). Jede RIA-Plattform hält heute Mehtoden für den API-Zugriff und die Verarbeitung der Inhalte (z.B. XML, JSON) bereit. RIAs werden i.a.R. nach dem MVC-Muster[15] entworfen, was eine Trennung in Präsentationsschicht, Geschäftslogik und Datenschicht bedeutet. Durch diese lose Kopplung lassen sich externe Services einfach in RIAs einbinden.

- Offline Anwendungen: RIAs müssen nicht zwingenderweise im Browser betrieben werden bzw. funktionieren zum Teil auch ohne Internetverbindung. Das Browser-Plugin Google Gears[16] z.B. schafft genau das für Webanwendungen und errichtet eine Brücke zwischen on- und offline Anwendungen. Dabei werden Daten bei Bedarf in einer lokalen Datenbank gespeichert und können so erst später mit einem Server synchronisiert werden [Google, 2009]. Adobe AIR ist eine weitere plattformunabhängige RIA-Laufzeitumgebung für den Desktop. AIR-Applikationen werden im Gegensatz zu Applikation, die Gears verwenden, nicht im Browser ausgeführt. [Ehrenstein, 2009].

- Plattformunabhängigkeit: Rich Internet Anwendungen sollen einfach auf verschiedene Plattformen und Endgeräte verteilt werden können. Skript-basierte RIAs (Stichwort AJAX) werden im Browser ausgeführt und verwenden das Web als Plattform (vgl. 2.1.1.2). Dazu braucht nur der jeweilige Browser auf den unterschiedlichen Betriebssystemplattformen unterstützt bzw. die Anwendung darauf getestet werden. Bei Plugin-basierten RIAs wie z.B. Flash, Sivlerlight oder JavaFX muss natürlich das Plugin selbst für die einzelnen Plattformen entwickelt bzw. portiert werden, um so eine Anwendung unabhängig von der Plattform laufen lassen zu können. (Dies ist natürlich nicht Aufgabe der Anwendungsentwickler, sondern der Plattformentwickler.) Neben PCs sollen RIAs auf verschiedene andere Endgeräte wie Pocket PCs, Smartphones und Handys einfach verteilt werden können. Bestimmte RIA-Plattformen stellen eine spezielle Plattformen für mobile Endgeräte zur Verfügung, so z.B. Flash Lite[17] oder JavaFX Mobile[18], welche auf der Java ME Plattform[19] basiert. Silverlight for Mobile[20] ist gerade in Entwicklung.

---

[15] Model-View-Controller
[16] `http://gears.google.com/`
[17] `http://www.adobe.com/products/flashlite/`
[18] `http://www.sun.com/software/javafx/mobile/`
[19] `http://java.sun.com/javame/`
[20] `http://silverlight.net/learn/mobile/`

### 3.1.4 RIA Technologien

Es wurden bereits verschiedene Plattformen und Technologien zur Umsetzung einer RIA erwähnt. Diese Technologien können in verschiedene Kategorien die Entwicklungs- und Laufzeitumgebung betreffend eingeteilt werden. Je nach Autor wird von drei (vgl. [Farrell and Nezlek, 2007]) oder vier (vgl. [Bozzon et al., 2006]) Kategorien gesprochen. Beide Autoren decken das gleiche Spektrum ab, wobei Farrel zwei Kategorien von Bozzon in eine zusammenfasst. Im Folgenden wird die feinere Einteilung nach [Bozzon et al., 2006] beschrieben:

- Skript-basierte RIA: Eine skript-basierte RIA verwendet eine Skript-Sprache für die Implementierung der Logik auf Client-Seite. Seit 2005 sind diese Webanwendungen unter dem Akronym AJAX[21] bekannt. Dazu wird eine Kombination der Technologien (X)HTML, CSS[22], DOM und JavaScript eingesetzt. Das Design und Layout des Interfaces wird mit CSS und HTML umgesetzt. Mit JavaScript werden asynchrone Anfragen an eine Webserver gesendet. Die Antwort wird mit JavaScript verarbeitet und das User Interface (kurz UI) mittels DOM-Manipulation aktualisiert [Garrett, 2005]. Aufgrund der verschiedenen Browser-Implementierungen der W3C-Spezifikationen von HTML und CSS kommt es teilweise zu unterschiedlichen UI-Darstellungen und Fehlverhalten in Anwendungen. JavaScript Bibliotheken und Frameworks wie YUI[23], MooTools[24] oder dojo[25] schaffen hier Abhilfe, berücksichtigen Browser-Unterschiede und sorgen für eine gleiche Darstellung in allen gängigen Browsern. Daneben bieten sie nützliche Funktionen, erweitern die JavaScript-Sprache und ermöglichen eine modulare, objektorientierte Entwicklung
  [Orchard et al., 2009].

- Plugin-basierte RIA: Plugin-basierte RIAs werden auf einer eigenen Plattform erstellt und sind ohne ein geeignetes Plugin im Browser nicht ausführbar. D.h., es muss das jeweilige Plugin im Browser installiert werden, was ein Nachteil dieser Lösungen ist. Bei einem Computer zu Hause stellt dies wahrscheinlich kein Problem dar, allerdings in Unternehmen mit strengen IT-Sicherheitsvorschriften und eingeschränkten Rechten u.U. schon. Plugin-basierte RIA-Plattformen sind z.B. Adobe Flex, Microsoft Silverlight oder JavaFX. Anwendungen dieser Plattformen werden in nativen Code übersetzt. Die Verarbeitungs- und Darstellungsgeschwindigkeit des nativen Codes ist i.a.R. schneller als bei skript-basierten RIAs, da Berechnungen des Layouts und Erscheinungsbildes nicht erst zur Laufzeit verarbeitet werden müssen, sondern schon zur Entwurfszeit durchgeführt wurden [Ehrenstein, 2009]. Diese RIAs

[21] Asynchronous JavaScript and XML
[22] Cascading Stylesheets
[23] `http://developer.yahoo.com/yui/`
[24] `http://mootools.net/`
[25] `http://www.dojotoolkit.org/`

bieten einige weitere Vorteile gegenüber skript-basierten RIAs und setzen dort an, wo HTML seine Grenzen hat. Einige dieser Vorzüge sind z.B. Drag-and-Drop, die einfache Integration von verschiedenen Multimediaformaten (Audio, hochauflösendes und Streaming Video sowie 3D Grafiken) oder die Verabreitung und Anzeige von Vektorgrafiken. Leistungsstarke Funktionen für die Datenvisualisierung (z.B. Diagramme, erweiterte Datengitter, Graphendarstellung usw.) sind ebenso Bestandteil wie umfangreiche Komponentenbibliotheken mit nützlichen, sofort zu verwendenden Komponenten wie z.B. Schiebereglern, Farbwählern uvw. Weiters stellen diese Plattformen ein Effekte- und Animationssystem zur Verwaltung von Bewegungen und Übergängen ebenso bereit sowie Möglichkeiten zur Erstellung von komplexen, adaptiven UI-Layouts. Die Entwicklung der Client-Logik erfolgt mittels einer objektorientierten Sprache wie .NET (Silverlight), Actionscript (Flex) oder Java (JavaFX); das User Interface wird hingegen meist mit einer deklarativen Sprache gestaltet (XAML, MXML, JavaFX Script) (vgl. [Microsoft, 2009], [Adobe, 2009]. [JavaFX, 2009]).

- Browser-basierte RIA: Diese verwenden eine auf XML aufbauende Sprache zur deklarativen Beschreibung von Benutzerschnittstellen (GUI). Mozilla XUL (XML User Interface Language) ist ein Beispiel für so eine Sprache. XUL wird z.B. nativ vom Browser Mozilla Firefox unterstützt und Mozilla bietet mit XULRunner auch eine Plattform für XUL-Anwendungen [Mozilla, 2009b]. XUL verwendet verschiedene W3C Standards wie CSS, DOM oder auch JavaScript. In XUL deklarierte GUIs ermöglichen eine Trennung von Layout und Design. Das Design wird per CSS festgelegt. So kann schnell der "Skin" einer Anwendung gewechselt werden, ohne das Layout oder die Logik ändern zu müsen. Browser-basierte RIAs stellen den geringsten Anteil an RIAs [Farrell and Nezlek, 2007].

- Web-basierte Desktop Technologien: Wenn Webanwendungen außerhalb des Browsers auf dem Desktop ausgeführt werden, so spricht [Bozzon et al., 2006] von web-basierten Desktop Technologien. Adobe AIR, Java Web Start[26] und Mozilla Prism[27] sind drei RIA-Plattformen, die Webapplikationen auf den Desktop bringen. Seit Version 3 ist auch Microsoft Silverlight zur Erstellung von Desktop Anwendungen geeignet (vgl. [Microsoft, 2009]). Diese Anwendungen laufen betriebssystemunabhängig als Desktopanwendungen in einem eigenen Prozess und können auch auf mehr Systemresourcen als herkömmliche, im Browser betriebene RIAs nutzen. Adobe AIR z.B. bietet eine Schnittstele zum lokalen Dateisystem, womit Dateien geöffnet, gespeichert und auch gelöscht werden können. Auch Drag-and-Drop von Dateien zwischen herkömmlichen Desktop- und AIR-Anwendungen sowie die Nutzung der Zwischenablage des Betriebssystems in AIR-Anwendungen ist möglich.

---

[26] `http://java.sun.com/javase/technologies/desktop/javawebstart/`
[27] `https://mozillalabs.com/prism/`

Auf die einzelnen Plattformen und Frameworks wird an dieser Stelle nicht näher eingegangen, da es den Rahmen der Arbeit sprengen würde. Für weiterführende Informationen sei hier auf die angegebenen Quellen verwiesen.

### 3.1.5 Die Architektur einer RIA

Es existieren nicht nur verschiedene RIA-Technologien, -Plattformen und dazugehörige Frameworks, sondern auch verschiedene Ausprägungen der Softwarearchitektur einer RIA. Auch wenn die Technologie einen Einfluss auf die Wahl der passenden Architektur einer RIA hat, so ist die Architektur vor allem von den gewählten Produkten und anwendungsspezifischen Entwurfsentscheidungen abhängig. In Abbildung 3.2 sind die verschiedenen Möglichkeiten der Architekturen zu sehen. Diese werden in diesem Abschnitt näher behandelt und beruhen auf [Walter, 2008] sowie [Domenig, 2005].

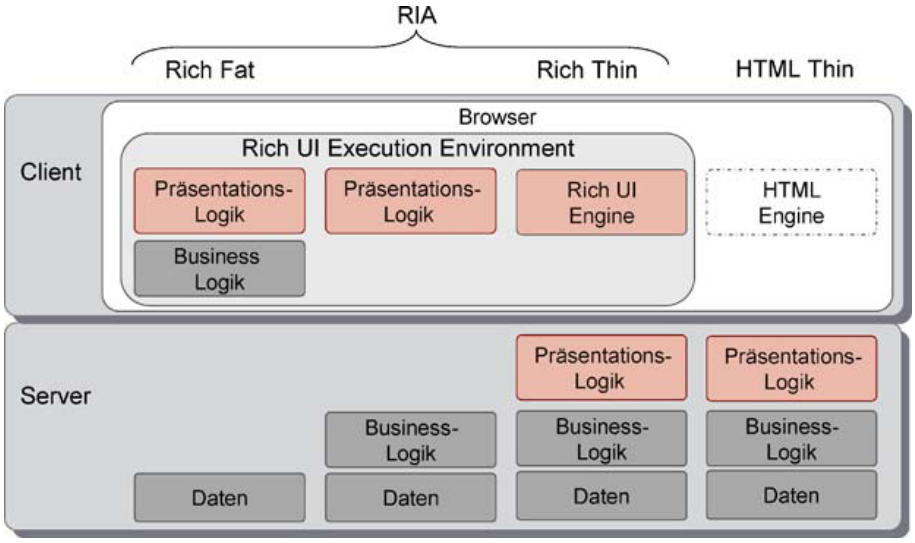



Abbildung 3.2: RIA-Architektur-Optionen (entnommen aus [Walter, 2008])

Softwarearchitekturen von RIAs und Webandwendungen werden typischerweise in drei Schichten (sog. Tiers) realisiert: Datenschicht, Geschäftslogik und Präsentationslogik. Die Datenschicht beinhaltet Strategien zum Zugriff auf und zur persistenten Haltung von Daten. Diese werden von der Geschäftslogik verarbeitet und die Präsentationslogik akutalisiert das GUI, sodass der Benutzer die Änderungen sieht. RIA-Architekturen reichen von "Rich Fat Clients" (in der Abbildung ganz links) bis zu "Rich Thin Clients" (in der Abbildung die zweite von links). Zum Vergleich ist in der Abbildung ganz rechts die Architektur einer herkömmlichen, klassischen HTML-basierten Webanwendung (HTML Thin Client) zu sehen. Dabei ist zu erkennen, dass alle drei Schichten am Server implementiert sind, und am Client nur ein Browser zum Anzeigen gebraucht wird. Die Architektur eines Rich Thin Clients ist

dieser Architektur sehr ähnlich, nur dass dabei eine mächtigere "Rich UI Engine" am Client zum Einsatz kommt. Vergleicht man diese Architektur mit den beiden auf der linken Seite, so ist zu sehen, dass mehr und mehr Logik auf den Client ausgelagert wird. So wird in einer Hybrid-Variante die Präsentationslogik auf den Client ausgelagert und bei einem Rich Fat Client sogar die Geschäftslogik dazu. In allen Varianten wird eine Laufzeitumgebung für die Interaktivität auf der Clientseite benötigt. Details dazu wurden bereits im vorigen Abschnitt behandelt.

Im Folgenden werden die Vorteile der jeweiligen Architekturvarianten erörtert. Rich Thin Clients zeichnen sich durch folgende Vorteile aus:

- Rich Thin Clients können ohne großen Aufwand implementiert werden, da sie in die bestehende Webarchitektur integriert werden können. Es muss nur die Präsentationsschicht durch eine reichhaltigeres GUI ersetzt werden. Geschäftslogik und Datenschicht müssen nicht geändert werden, da sie sich nach wie vor am Server befinden. Da sich alle Schichten am Server befinden, muss sich der Entwickler nicht mit Problemen der verteilten Programmierung auseinandersetzen, was eine Vereinfachung der Entwicklung und Wartung der Anwendung mit sich bringt.
- Durch die Verwendung einer generischen "UI Rendering Engine" wird die Anwendung weiterhin auf allen Umgebungen laufen, für welche diese "Rendering Engine" vorliegt.
- Sofern die "Rendering Engine" am Client verfügbar ist, muss für die Ausführung der Anwendung nichts weiter heruntergeladen werden.
- Die Auslagerung von Anwendungscode auf den Client hat zur Folge, dass der RIA-Provider die Kontrolle über die verwendeten Softwareversionen verliert. Solange eine RIA verwendet wird, die im Browser läuft, ist dies noch kein Problem, da jeweils die aktuellste Version zu Programmstart heruntergeladen und ausgeführt wird. Bei Desktop-basierten RIAs, die eine Installation voraussetzen (z.B. Adobe AIR) und Teile der Geschäftslogik beinhalten, muss auf Abwärtskompatibilität und u.U. multiversionsfähige Schnittstellen geachtet werden. Denn User können nicht gezwungen werden, die neueste Version der Clientsoftware zu installieren und damit kompatible Clientsoftware zu haben.
- Es tun sich keine neuen Sicherheitslücken am Client auf, da die Anwendung nach wie vor am Server gehalten wird.

Rich Fat Clients bieten folgende Vorteile:

- Die Integration von Fat Clients in die native Desktopumgebung ist einfacher und homogener. Zum Beispiel kann mit anderen Clientanwendungen kommuniziert, auf das lokale Dateisystem und die Peripherie zugegriffen oder auf Ereignisse des Betriebssystems reagiert werden.

- Durch die Auslagerung von Applikationslogik auf den Client werden die Ressourcen des lokalen Rechners in Anspruch genommen. Dies wirkt sich positiv auf die Skalierung des Gesmatsystems aus.

- Wird mehr Code und damit mehr Berechnungen auf den Client abgewälzt, so reduziert das nicht nur die Server- sondern auch die Netzwerklast. Dies hat auch eine effizientere Skalierbarkeit der Server zur Folge. In extremen Fällen kann sogar auf einen Applikationsserver verzichtet und nur ein Datenbankserver verwendet werden.

Die Vorteile von Richt Thin Clients sind insbesondere bei einer großen Anzahl von Benutzern zu spüren. Dies stellt somit auch hohe Skalierbarkeitsanforderungen an die Server, womit der Vorteil von Rich Fat Clients nicht zum Tragen kommt. Eine "goldene Regel" für einen Rich Thin oder Fat Client lässt sich somit nicht definieren, die Wahl der Architektur ist sehr kontextbezogen. Allerdings ist nach [Walter, 2008] eine zentralisierte, möglichst serverseitige Lösung in den meisten Fällen einer hoch dezentralierten vorzuziehen.

## 3.2 Konzept

In Kapitel 1.1.2 wurden zwei Szenarien vorgestellt, wie der Anwendungsfall umgesetzt werden kann, um die vorhandenen strukturierten Informationen durch Dritte anzureichern. Beide Anwendungsszenarien wurden im Rahmen des in Kapitel 1.1 bereits erwähnten Auftrages der SUCCON an die TU Graz zur Durchführung einer Feasibility Studie konzipiert, das zweite der beiden Anwendungsszenarien im Rahmen dieses Auftrages auch umgesetzt. Es folgt nun die Beschreibung dieses Konzepts, das die Zusammenfassung der Anforderungen aus dem Anwendungsszenario sowie die Umsetzung dieser beinhaltet.

### 3.2.1 Anforderungen

Aus der Beschreibung des Szenarios lassen sich folgende Anforderungen ableiten:

- Die Anwendung ist webbasiert und wird im Browser ausgeführt.

- Es sollen nach Möglichkeit Open Source oder frei verfügbare Bibliotheken, Frameworks und Komponenten eingesetzt werden.

- Benutzer können nach Verbindungen durch Eingabe eines Texts suchen.

- Die Suchresultate müssen grafisch so aufbereitet werden, dass die Richtung und die Stärke der Wirkungsbeziehung für den Benutzer eindeutig erkennbar sind.

- Die Ergebnisse sollen möglichst von verschiedenen Erfolgsprofilen stammen, da es von Bedeutung ist, nahezu alle Erfolgsprofile mit weiteren Daten anzureichern.
- Benutzer können die gefundenen Verbindungen mittels fünfstufiger Skala in beide Richtungen bewerten.
- Durch einen Benutzer bewertete Verbindungen sollen beim Finden der Resultate miteinfließen.
- Anonyme (nicht angemeldete) Benutzer können nur eine begrenzte Anzahl an Suchen absetzen. Ist dieses Limit erreicht, müssen sie sich anmelden (und u.U. vorher registrieren), um den Dienst weiter nutzen zu können.
- Es können neue Benutzer angelegt werden. Dazu ist die Angabe von Benutzername und Passwort notwendig. Der Username muss im ganzen System eindeutig sein.
- Das Passwort soll verschlüsselt gespeichert werden.
- Benutzer können sich mit Benutzername und Passwort anmelden und sich vom System abmelden.
- Angemeldete Benutzer haben keine Beschränkung, was die Anzahl der Suchen betrifft.
- Es sollen möglichst viele bestehende Verbindungen von den Benutzern bewertet werden.

Die SUCCON hat bereits ein Datenbankschema entwickelt, in dem abgebildet ist, wie die strukturierten Daten zur Speicherung von Erfolgsdiagnosen abgelegt sind. Um einen besseren Überblick der anzureichernden Daten zu erhalten, wird ein Auszug des existierenden Datenbankmodells mit den wichtigsten Entitäten und Relationen zueinander in Abbildung 3.3 gezeigt. Die drei zentralen Entitäten Erolgsprofil (`successprofile`), Erfolgsfaktor (`successfactor`) sowie Erfolgsverbindung (`sfconnection`) wurden bereits ausführlich in Kapitel 1.1.1 beschrieben. Zur Stammdatenverwaltung sind bis dato die zwei Entitäten Benutzer und Organisation vorhanden. Diese werden nachfolgend näher beschrieben.

- **Benutzer**: Zur Identifikation von Benutzern wird diese Entität verwendet. Benutzerinformationen wie Benutzername und Passwort werden darin gespeichert. Ein Benutzer wird einer Organisation zugeordnet.
- **Organisation**: Eine Organisation ist das Bindeglied zwischen Erfolgsprofilen und Benutzern. Ein Erfolgsprofil wird nicht direkt einem Benutzer, sondern einer Organisation zugeordnet. Der Grund dafür liegt darin, dass der Betrachtungsgegenstand einer Erfolgsdiagnose i.A. im organisationalen Umfeld liegt

und somit mehrere Personen involviert sind. Dabei können einer Organisation mehrere Benutzer zugeordnet werden. Es können auch mehrere Erfolgsprofile zu einer Organisation angelegt werden, da Erfolgsdiagnosen zu verschiedenen Betrachtungsgegenständen durchgeführt werden können.

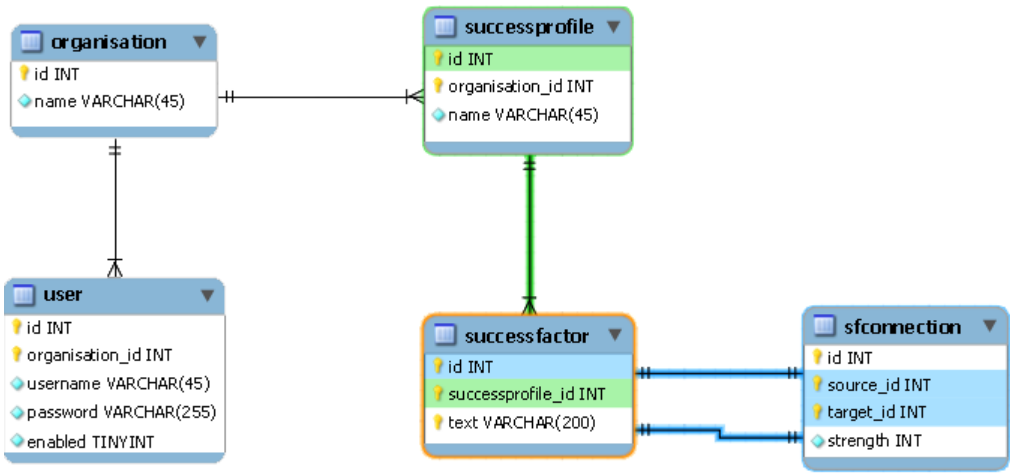



Abbildung 3.3: Auszug aus dem existierenden Datenbankmodell

### 3.2.2 Umsetzung der Anforderungen

Es wurde eine Webanwendung entwickelt, mit der Benutzer nach bestehenden Erfolgsverbindungen suchen und diese bewerten können. Aus den angeführten Anforderungen kristallisieren sich zwei Bereiche von zentraler Bedeutung heraus: Die Suche und die Bewertung von Erfolgsverbindungen. Diese zwei Bereiche beinhalten die eigentliche Geschäftslogik der Anwendung. Daneben gibt es noch folgende Support- und Infrastruktur-Bereiche: das Usermanagement, Sicherheit (Authentifizierung, Autorisierung) und die Datenpersistierung (Datenbank). Es wird nun das entwickelte Konzept beschrieben, das darauffolgende Kapitel (4) behandelt die technische Implementierung.

#### 3.2.2.1 Suche

Durch Eingabe eines Texts in ein Suchtextfeld kann mit der Webanwendung nach Erfolgsverbindungen in existierenden Erfolgsprofilen gesucht werden. Die Suchergebnisse werden in Listenform untereinander dargestellt. Dabei ist auf der linken und auf der rechten Seite einer Suchergebnisreihe je ein Erfolgsfaktor zu sehen. Dazwischen wird über einen Pfeil angezeigt, welcher Erfolgsfaktor auf welchen wirkt. Handelt es sich um eine bidirektionale Beziehung, so wird diese in der selben Reihe

mit einem Pfeil mit zwei Spitzen (in beiden Richtungen) angezeigt. Der Benutzer hat die Möglichkeit bzw. ist angehalten, die ihm präsentierten Verbindungen über einen fünfstufigen Bewertungsbalken zu bewerten. D.h., es wird eine Suchmaschine mit dem Ziel umgesetzt, Wirkungsbeziehungen zwischen Erfolgsfaktoren von bestehenden Erfolgsprofilen mit weiteren Daten anzureichern, um so eine andere (externe) Sichtweise auf diese (über Bewertungen) zu erhalten. Um dieses Ziel zu erreichen, darf der Faktor Benutzermotivation nicht außer Acht gelassen werden. Denn es ist nicht zu erwarten, dass Benutzer von sich aus viele Bewertungen abgeben. Sie werden vielmehr die Suche nutzen, um ihr Informationsbedürfnis zu befriedigen. Deswegen muss ein Weg gefunden werden, Benutzer durch einfache Beschränkungen bzw. Aufforderungen zu Bewertungen zu motivieren. Das Regelsystem darf also nicht zu restriktiv sein und pro Suche viele Bewertungen verlangen, da die Benutzer die Webanwendung nicht verwenden würden. Restriktionen dürfen aber auch nicht komplett fehlen oder zu lose sein, da ansonsten u.U. zu wenige Verbindungen bewertet würden.

Aus diesen Gründen wurde folgendes Konzept zur Suche und Bewertung entwickelt: Verwendet man das System als anonymer, unangemeldeter User, so sind grundsätzlich drei Suchen frei. Danach ist eine Registrierung notwendig, um den Dienst weiter nutzen zu können. Eine Registrierung ist i.A. kostenfrei und hat den Vorteil, dass eine personalisierte Suche ermöglicht wird. Es kann für jeden Benutzer eine Historie dessen Suchen sowie dessen Bewertungen erstellt werden, welche in die Erstellung der Suchresultate einfließt. Bewertungen können so einem dezidierten User zugeordnet werden, anstatt einem anonymen Benutzer, was auch für die Erstellung von Statistiken aufschlussreicher ist.

Ist man als registrierter Benutzer angemeldet, so können beliebig viele Suchen abgesetzt werden, nur muss nach jeder dritten Suche mindestens ein Verbindungspaar bewertet werden, um das System mit weiteren Daten anzureichen und einen Nutzen zu erzielen. Optional wäre folgende Erweiterung vorstellbar: Nach einer bestimmten Anzahl an abgesetzten Suchen sind mehr Bewertungen pro Suche (z.B. zwei) notwendig, um weitersuchen zu können.

Die Suche basiert auf einem textuellen Vergleich des Suchtexts mit den Quell- und Zielerfolgsfaktoren der Erfolgsverbindungen. Alle Erfolgsverbindungen, in deren Quell- oder Zielerfolgsfaktoren der eingegebene Suchtext enthalten ist, gehören zur Menge der potenziellen Ergebnisse. Doch es werden pro Suche nur drei Suchresultate präsentiert, auch wenn mehr Ergebnisse gefunden werden. Zwei zentrale Anforderungen sind, dass die Daten möglichst aller Erfolgsprofile und insgesamt möglichst viele einzelne Erfolgsverbindungen durch Dritte bewertet werden. Um diese Anforderungen zu erfüllen, wurden verschiedene Maßnahmen getroffen. In Kombination mit der Beschränkung, dass ein angemeldeter Benutzer nach jeder dritten Suche bewerten muss (um weitere Suchen absetzen zu können), werden in Summe mehr

Bewertungen abgegeben, als wenn alle Ergebnisse zu einem Suchbegriff präsentiert würden. Denn für weitere Ergebnisse zu einem Suchbegriff muss eine neue Suche gestartet werden. So muss ein Benutzer in weiterer Folge früher bewerten, um wieder suchen zu können. Für eine breitere Streuung werden die Resultate randomisiert von verschiedenen Erfolgsprofilen entnommen. Zusätzlich wird die bisherige Bewertungshistorie eines Benutzers berücksichtigt: Dem Benutzer werden keine Ergebnisse präsentiert, die er innerhalb der letzten 24 Stunden bewertet hat, da der Mehrwert beschränkt und die Bewertung nicht repräsentativ ist, wenn ein Benutzer dieselbe Verbindung in einem kurzen Zeitraum (innerhalb von 24 Stunden) zwei Mal verschieden bewertet. Ein anderer Vorteil, den die beschränkte Menge an Suchresultaten mit sich bringt, ist, dass die Benutzer durch die neue und (anfangs) u.U. etwas ungewöhnliche Darstellung der Resultate nicht überfordert werden und die präsentierten Informationen rezipieren können. Eine geringe Anzahl von Resultaten wird vermutlich eher beachtet bzw. wahrgenommen und infolgedessen auch bewertet. Bei einer Anzahl von etwa zehn Resultaten, ist es wahrscheinlicher, dass der Benutzer gewisse Resultate einfach überlist und so auf mögliche Bewertungen vergisst.

#### 3.2.2.2 Bewertung

Es wurde bereits erwähnt, dass Bewertungen in die Auswahl der Suchresultate miteinfließen. Mit Hilfe der Bewertungen gibt der Benutzer seine Meinung ab, wie er den Wirkungszusammenhang zwischen zwei Erfolgsfaktoren sieht. In Kapitel 1.1.1 wurden Erfolgsverbindungen bereits eingehend behandelt. Der Vollständigkeit halber seien hier noch einmal die wichtigsten Punkte erwähnt:

- Eine Verbindung bezeichnet einen gerichteten Wirkungszusammenhang zwischen zwei Erfolgsfaktoren, d.h., dass sich ein Erfolgsfaktor auf einen anderen auswirkt.
- Die Wirkung hat eine positive oder negative Polarität, ein Erfolgsfaktor wirkt sich also positiv oder negativ auf einen anderen aus. Dies wird durch ein "+" oder ein "−"symbolisiert.
- Ein Wirkung hat eine Intensität, die stark oder schwach sein kann. Eine starke Intensität wird durch ein doppeltes Symbol (zwei Zeichen), eine schwache durch ein einfaches Symbol (ein Zeichen) dargestellt.
- Die Wirkung wird als eine Kombination aus Polarität und Intensität dargestellt: Somit wird eine schwach positive Wirkung z.B. durch "+", eine stark negative durch "−−" dargestellt. Es gibt somit grundsätzlich vier verschiedene Werte, die eine Wirkungsbeziehung bzw. Erfoglsverbindung haben kann.
- Setzt ein Benutzer eine Suchanfrage ab, so werden ihm existierende Erfolgsverbindungen präsentiert. Ein Benutzer kann allerdings der Ansicht sein, dass kein Wirkungszusammenhang zwischen zwei Erfolgsfaktoren besteht. Ist das

der Fall, so kann er dies durch Klicken auf "0" im Rating-Balken tun. Somit gibt es einen fünften Bewertungswert.

- Bidirektionale Bewertungen sind auch möglich. D.h., dass pro Erfolgsfaktoren-Paar zwei Bewertungen abgegeben werden können (A auf B, B auf A).

Tabelle 3.1 zeigt zur Übersicht die fünf verschiedenen Werte und Symbole, die eine Wirkungsbeziehung haben kann:

| **Wert** | **Symbol** | **Bezeichnung** |
|---|---|---|
| 0 | −− | stark negativ |
| 1 | − | schwach negativ |
| 2 | 0 | keine Wirkung |
| 3 | + | schwach positiv |
| 4 | ++ | stark positiv |

Tabelle 3.1: Die Werte einer Wirkungsbeziehung

Es werden pro Erfolgsverbindung, die in den Suchresultaten präsentiert werden, zwei fünfstufige Balken zur Bewertung verwendet. Die Bewertung wird gespeichert, sobald auf einen der Buttons eines Bewertungsbalkens gedrückt wird. Diese Bewertung ist nicht endgültig: Der Benutzer hat die Möglichkeit, die Bewertung zu ändern, indem er auf einen anderen Button desselben Bewertungsbalkens klickt. Denn der Benutzer kann seine Meinung ändern, vorschnell oder unabsichtlich den falschen Button angeklickt haben. Wird eine abgegebene Bewertung geändert, so wird die alte, gerade abgegebene Bewertung überschrieben. Denn verschiedene Bewertungen zu einer Erfoglsverbindung innerhalb einer Zeitspanne von wenigen Sekunden von ein und demselben User haben wenig Aussagekraft und lassen darauf schließen, dass er anscheinend unsicher in seiner Meinungsvergabe war. Wie im vorigen Abschnitt zur Suche beschrieben, kann ein Benutzer erst nach 24 Stunden wieder das gleiche Suchresultat erhalten. Und erst nach dieser Zeitspanne kann er tatsächlich eine neue Bewertung abgeben, da er eventuell am nächsten Tag eine andere Sicht auf die Verbindung hat. Selbst wenn er die gleiche Bewertung wie am Vortag abgibt, wird die bestehende Bewertung nicht überschrieben, sondern eine neue angelegt. So erhält eine Bewertung ein größeres Gewicht, ebenso wenn verschiedene Benutzer eine Bewertung zu einer Erfolgsverbindung abgeben. Je mehr Bewertungen zu einer Verbindung abgegeben werden, desto mehr Aussagekraft haben diese, vor allem wenn sie von vielen verschiedenen Nutzern stammen (vgl. "Kollektive Intelligenz" in 2.1.1.2). Einfache statistische Auswertungen wie z.B. der Mittelwert der Bewertungen zu einer Erfolgsverbindung geben Aufschluss über die externe Sicht auf die Verbindungen innerhalb eines Erfolgsprofils und damit für den bzw. die Ersteller. Diese Auswertungen sind allerdings nicht Teil der Arbeit.

#### 3.2.2.3 Benutzermanagement

Für das Usertracking und um Bewertungen Benutzern zuordnen zu können, ist ein Benutzermanagement unumgänglich. Wenn jemand die Anwendung öffnet, verwendet er das System als anonymer Benutzer. Dies wirkt sich einerseits auf die Suchresultate als auch auf Bewertungen aus. Auch ein anonymer Benutzer kann Verbindungen bewerten. Tut er dies, wird so eine Bewertung mit einer eindeutigen systemweiten anonymen ID gespeichert (Diese ID ist für alle anonymen Benutzer gleich). Für anonyme Benutzer kann keine Historie der bisherigen Bewertungen erstellt werden, da verschiedene Benutzer gleichzeitig anonym das System verwenden können. Dies bedeutet, wenn ein anonymer Benutzer eine Suche absetzt, fließen die anonymen Bewertungen nicht in die Auswahl der Suchresultate. Sie verkleinern die Auswahl der Ergebnisse also nicht, da nicht festgestellt werden kann, von welchem anonymen Benutzer die Bewertung stammt. Somit können in den Suchresultaten anonymer Benutzer auch Erfolgsverbindungen enthalten sein, die sie zuvor bewertet haben. Wie im vorigen Abschnitt erklärt, ist dies für angemeldete User nicht möglich.

Ein Benutzer kann sich mit Benutzername und Passwort über einen eigenen Link registrieren. Weitere Informationen werden nicht abgespeichert und sind für die Anwendung im Rahmen der Arbeit nicht notwendig. Auch stehen dem Benutzer keine Funktionen zur Änderung der Benutzerinformationen wie Passwort zur Verfügung. Das Passwort wird verschlüsselt gespeichert. Hat ein Benutzer das Passwort vergessen, gibt es keinen Weg, das Passwort wieder zu herauszufinden oder per Email zu erhalten. In einer späteren Version der Anwendung werden solche Funktionen inklusive einer Stammdatenverwaltung je nach Bedarf bereitgestellt.

Hat ein anonymer Benutzer drei Suchen abgesetzt, so erscheint automatisch eine Warnung und das Registrierungsfenster, das er nicht schließen kann. So wird eine Registrierung erzwungen. In weiterer Folge kann er sich natürlich anmelden und das System (als angemeldeter und nicht mehr als anonymer User) verwenden.

Grundsätzlich sind zwei verschiedene Benutzergruppen voneinander zu unterscheiden: Erstens jene Gruppe, die Erfolgsprofile erstellt und somit die zu bewertenden Daten bereitstellt. Die Funktionalität des Erstellens von Erfolgsprofilen wird allerdings nicht im Rahmen der Masterarbeit entwickelt. Für die Entwicklung des Prototyps werden nur Testdaten erstellt. Die zweite Benutzergruppe sind die externen Benutzer, welche die Daten bewerten. Diese haben nur über die Webanwendung bzw. die Suche aufgrund der Geheimhaltung Zugriff auf die strukturierten Daten. Dies ist auch die einzige Gruppe, die im Rahmen der Entwicklung des Prototyps mit dem System interagieren wird. D.h., wird ein neuer Benutzer angelegt, so wird dieser der Gruppe “Extern” zugeordnet. Jedoch wird das System bereits in Hinblick auf einen möglichen realen Einsatz entworfen.

# Kapitel 4

# Implementierung

In diesem Kapitel wird die technische Umsetzung des entwickelten Konzepts beschrieben. Die Webanwendung erhielt den Codenamen OpenSearch. Dies ist darauf zurückzuführen, dass es sich dabei um eine für alle Benutzer offen zugängliche Suchmaschine handelt. Mit dieser kann nach unternehmenskritischen, geheimen Daten gesucht werden. Diese können auch bewertet werden. Zuerst wird ein Überblick über die RIA-Architektur gegeben und die verwendeten Technologien beschrieben. Danach werden das Datenbankschema, die Komponenten des Servers und Clients erläutert. Am Ende werden Screenshots der Anwendung gezeigt, um sich ein Bild vom graphischen User Interface zu machen.

## 4.1 Anwendungsarchitektur

Dieses Kapitel gibt einen Überblick über die Softwarearchitektur der implementierten RIA. Details werden hier noch nicht betrachtet, stattdessen werden die High-Level Architektur (Schichtenmodell) sowie die eingesetzten Technologien und die Gründe für deren Wahl erörtert. In den darauffolgenden Kapiteln werden die Schichten im Detail beschrieben.

### 4.1.1 High-Level Architektur

Es wurde eine Rich Internet Application mit einer Rich Thin Client-Architektur umgesetzt. Rich Thin Clients zeichnen sich dadurch aus, dass sich nur eine komplexe Präsentationsschicht auf dem Client befindet und keine Geschäftslogik. Diese beinhaltet der Server (vgl. Kapitel 3.1.5).

Der Grund für diese Wahl der Architektur war, dass auf einen zentralen Pool an unternehmenskrititschen Daten zugegriffen wird, der durchsucht wird und durch die User angereichert werden soll. Dieser gesamte Pool müsste bei einer Rich Fat Client Lösung an den Client gesendet werden, um die Suchresultate clientseitig zu erstellen.

Natürlich würde so Last vom Server genommen werden, allerdings müssten die Daten ständig mit dem Server sowie allen anderen verbundenen Clients synchronisiert werden, da dauernd Bewertungen abgegeben werden. Auch aufgrund der Geheimhaltung wäre eine clientseitige Lösung ein Problem. Die Geschäftslogik ist also am Server, welcher die Suchanfrage verarbeitet, die Resultate findet und diese an den Client sendet, welcher diese interpretiert, verarbeitet und darstellt. Die einzelnen Schichten der Architektur inklusive verwendeter Technologien sind in Abbildung 4.1 zu sehen.

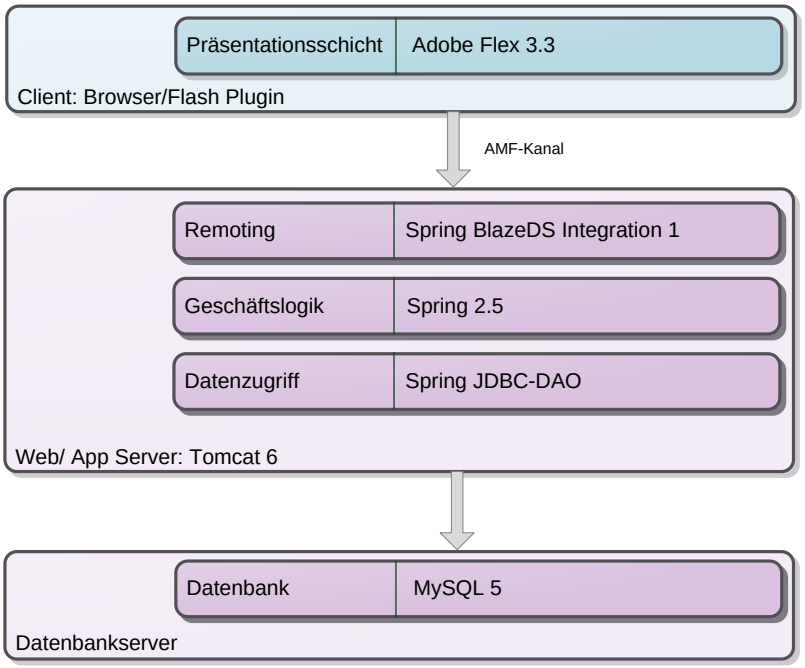



Abbildung 4.1: Die Schichten der RIA-Architektur

### 4.1.2 Eingesetzte Technologien

#### 4.1.2.1 Präsentationsschicht: Adobe Flex

Als Präsentationsschicht (Client) und somit RIA-Plattform wurde die Flash-Plattform respektive das Adobe Flex Framework gewählt. Die Flash-Laufzeitumgebung ist ein Browser-Plugin, das zwar separat installiert werden muss. Allerdings ist das Flash-Plugin, das in der Version 9 zur Ausführung von Flex Programmen notwendig ist, mit einer Durchdringung von 99% das am weitest verbreitete Plugin für RIAs bzw. reichhaltige Webinhalte [Adobe, 2010]. Durch das Plugin sind Flex-Anwendungen Browser-unabhängig und müssen im Gegensatz zu AJAX-Anwendungen nicht auf verschiedenen Browsern getestet werden müssen, um eine richtige Funktionsweise

und Darstellung in jedem Browser zu gewährleisten.

Adobe ist der Entwickler der Flash Plattform, die seit 1996 existiert und seitdem ständig weiterentwickelt wurde. Es handelt sich also um eine ausgereifte Technologie und mit dem Flex Framework wird seit März 2004 ein RIA-Framework angeboten, das mittlerweile in der vierten Version existiert. Zum Vergleich sind die RIA-Plattformen Microsoft Silverlight (erster Release im Dezember 2006 [Microsoft, 2010]) und Sun JavaFX (Version 1.0 im Dezember 2008 veröffentlicht [Sun Microsystems, 2010]) relativ junge Plattformen bzw. Frameworks.

Ein weiterer Grund für die Wahl des Flex Frameworks war, dass es einen großen Pool an vorhandenden UI- und Kommunikations-Komponenten gibt. So musste z.B. der Rating-Balken nicht selbst erstellt werden, sondern es konnte die existierende Flex-Komponente ToggleButtonBar verwendet und um zusätzliche Funktionalität erweitert werden. Zudem ist eine rege Community vorhanden, welche kostenlos nützliche und zum Großteil Open Source Komponenten und Bibliotheken anbietet. So wurde auch auf verschiedene weitere Frameworks zurückgegriffen, welche für die Implementierung notwendige Funktionalitäten abdecken, die das Flex Framework standardmäßig nicht zur Verfügung stellt.

#### 4.1.2.2 Geschäftslogik: Java, Spring

Auf der Serverseite wurde Java eingesetzt, da sich die Java Technologie auf der Serverseite mit Flex als Client sehr gut integrieren lässt. Natürlich ist Flex von der Server-Plattform unabhängig - es können auch die Sprachen PHP, ColdFusion oder die Technologie .NET eingesetzt werden. Für Java sprechen allerdings folgende Vorteile:

- Java ist im Enterprise und Webanwendungsbereich weit verbreitet.
- Es existieren viele Java Open Source Application Frameworks sowie Bibliotheken, die hilfreiche Komponenten zur Verfügung stellen ("Das Rad nicht neu erfinden").
- Adobe bietet mit BlazeDS bereits einen Open Source Remoting und Messaging Server in Java an, der die Kommunikation zwischen Java und Flex per JMS sowie RPC ermöglicht.

Weitere Vorteile für eine Kombination von Flex und Java sind nach [Marx, 2009]:

- Auffallende Ähnlichkeit zwischen Java und ActionScript[1] was Spracheigenschaften, Konzepte und Syntax betrifft.
- ActionScript ist (wie Java) eine objektorientierte und typisierte Sprache.

[1] ActionScript ist die Programmiersprache hinter Flex.

- Die Lernkurve für Java Kenner ist gering.
- BlazeDS ermöglicht standardmäßig die Verwendung des binären Datenaustauschformats AMF[2] zur Datenübertragung bzw. Objekt-Serialisierung zwischen ActionScript und Java. Das bringt Performance Vorteile gegenüber einer Übertragung in XML mit sich und vereinfacht die Programmlogik, da der Overhead des XML-Parsens und in weiterer Folge die Erstellung der Daten-Objekte entfällt.

Als Applikations Framework und für die Geschäftslogik wurde das Spring Framework eingesetzt. Dies ist ein "Lightweight Container/-Framework mit Dependency Injection", das die Konfiguration und Zusammenstellung komplexer Anwendungen aus einfacheren Komponenten ermöglicht. Es stellt eine umfassende Infrastrukturfunktionalität wie Web Services, Transaktionsverwaltung, Sicherheitsmechanismen, Persistenz-Frameworks, uvw. zur Verfügung (vgl. [Walls and Breidenbach, 2008]).

Für den Datenzugriff wurden Spring JDBC-DAOs[3] verwendet. JDBC[4] ist eine einheitliche Datenbankschnittstelle der Java-Plattform. Als Web Server bzw. Servlet Container wurde Apache Tomcat[5] eingesetzt.

#### 4.1.2.3 Remoting: Spring BlazeDS Integration

Zur Kommunikation zwischen Client (Flex) und Server (Java) wurde BlazeDS bzw. das Spring BlazeDS Integration Modul verwendet. BlazeDS[6] ist ein Java Remoting und Messaging Server, um von Flex Anwendungen auf Daten auf einem Server zugreifen zu können. Spring BlazeDS Integration[7] erlaubt eine enge und naürliche Integration von BlazeDS in Spring. Dabei übernimmt Spring die Konfiguration von BlazeDS.

#### 4.1.2.4 Datenschicht: MySQL

Als Datenspeicher für persistente Anwendungsdaten wird die Datenbank MySQL 5 verwendet. MySQL[8] ist das meist verbreitete Open Source Datenbankmanagementsystem, bietet einen großen Funktionsumfang und ist für alle Plattformen verfügbar. Alternativ häte PostgreSQL eingesetzt werden können. Diese beiden gelten als die umfangreichsten und beliebtesten Open Source Datenbanken und werden in sehr vielen professionellen Webprojekten und Enterprise Anwendungen eingesetzt (vgl. [Holdener, 2008]).

---

[2] Action Message Format
[3] Data Access Objects
[4] Java Database Connectivity
[5] `http://tomcat.apache.org/`
[6] `http://opensource.adobe.com/wiki/display/blazeds/`
[7] `http://www.springsource.org/spring-flex`
[8] `http://www.mysql.com/`

Die folgenden Abschnitte enthalten Implementierungsdetails zu den einzelnen logischen Einheiten Datenbank, Server und Client.

## 4.2 Datenbankmodell

Es bestand bereits ein Datenbankschema, welches die strukturierten, anzureichernden Daten beschreibt. Dieses ist in Abbildung 3.3 zu sehen und wurde im Kapitel 3.2.1 beschrieben. Das Datenbankschema wurde verändert und erweitert, um die geforderten Funktionalitäten zu erfüllen. Es ist in Abbildung 4.2 auf der folgenden Seite zu sehen.

Das Schema ist in drei logische Bereiche unterteilt: `StructuredData`, `OpenSearch` und `UserManagement`. `StructuredData` beinhaltet die existierenden, strukturierten Daten. Dies umfasst die drei Tabellen `successprofile`, `sfconnection` und `successfactor`. Während die ersten beiden Tabellen unverändert blieben, wurde die Tabelle `successfactor` verändert. Diese speichert Erfoglsfaktoren und enthielt in der existierenden Version u.a. den Text bzw. Namen eines Erfolgsfaktors. Dieser Text wurde herausgelöst und wird nun in der Tabelle `sftext` gespeichert. So sollen Redundanzen vermieden werden, da die Texte "unique" gespeichert werden. D.h., dass nicht zwei Mal der gleiche Text in der Tabelle vorhanden ist. So wird nur die ID des referenzierten Texts (Fremdschlüssel `sftext_id`) in `successfactor`, anstatt der Text selbst gespeichert. Damit sind Suchen in weiterer Folge einfacher, und es kann ein gemeinsames Vokabular (beim Erstellen eines Erfolgsprofils) aufgebaut werden.

Die Tabelle `sftext` gehört neben `sftextconnection` und `usersftextconnectionrating` zum logischen Bereich `OpenSearch`, welcher die Tabellen der Geschäftslogik beinhaltet. Die Bewertungen der externen Benutzer werden nicht auf Ebene der Erfolgsfaktoren (und Erfolgsverbindungen), sondern auf Ebene der Texte selbst gespeichert. Denn für den Benutzer ist es nicht wichtig, zu welchem Erfolgsprofil eine Erfolgsverbindung gehört, sondern welche Texte in Beziehung zueinander stehen. Vielmehr darf er aufgrund der Restriktion der Geheimhaltung gar nicht erfahren, zu welchem Erfolgsprofil eine Verbindung gehört. Die Erfolgsverbindungen der externen User werden in `sftextconnection` gespeichert, welche aus ID, Quell- und Zieltext (beides Fremdschlüssel) besteht. Die Bewertung selbst ist in der Tabelle `usersftextconnectionrating` gespeichert, welche einen Benutzer und eine Textverbindung verknüpft und die abgegebene Bewertung enthält.

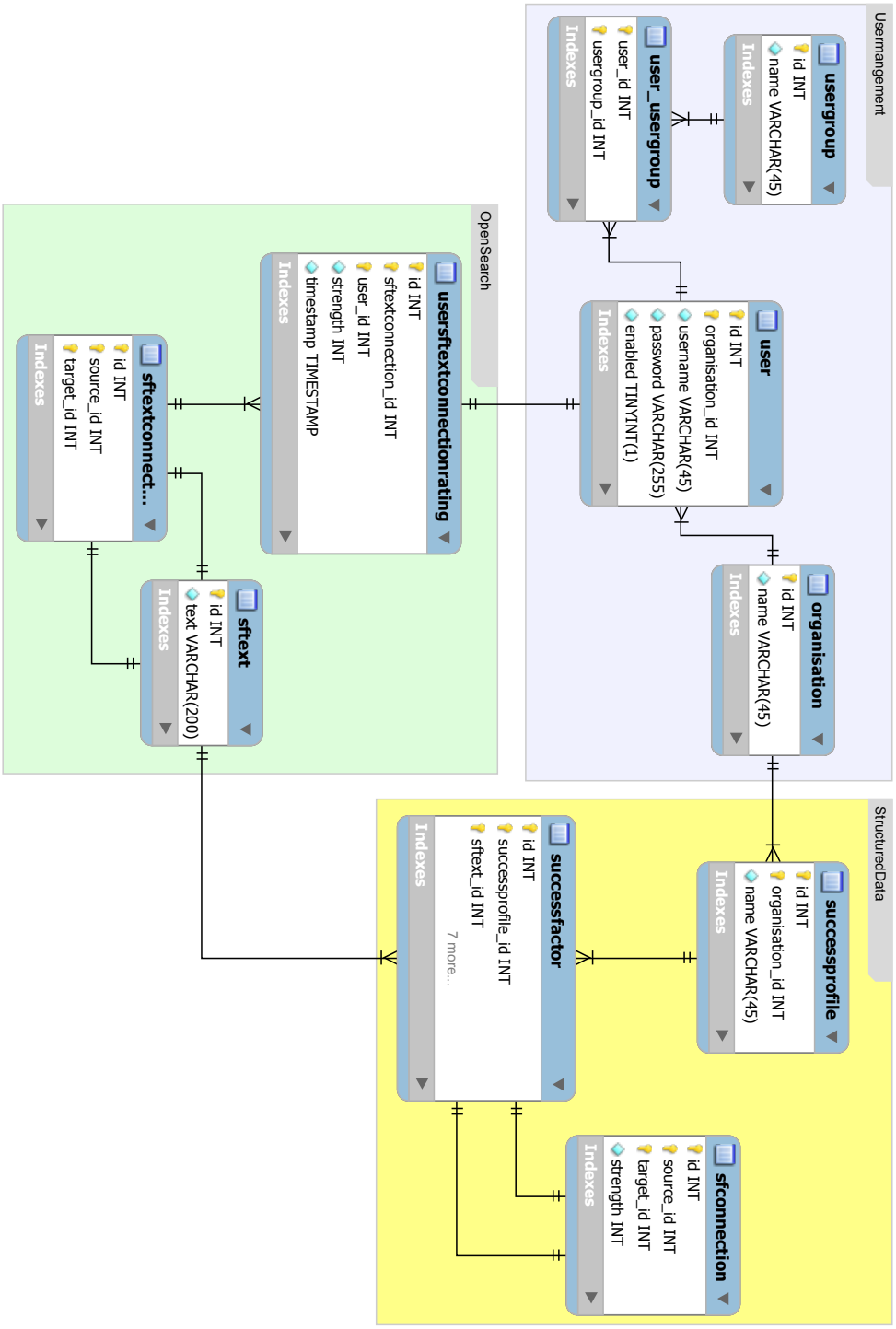



Abbildung 4.2: Datenbankschema der Webanwendung

Der dritte logische Bereich ist das `UserManagement`. Dazu zählen zusätzlich zu den zwei existierenden Tabellen `user` und `organisation` die beiden Tabellen `usergroup` und `user_usergroup`. `usergroup` enthält die verschiedenen Benutzergruppen (bzw. Benutzerrollen) und über die Zwischentabelle `user_usergroup` werden die Benutzer mit Benutzergruppen verknüpft. Diese n:m-Beziehung impliziert also, dass ein Benutzer mehreren Benutzergruppen angehören kann. Diese Tabellenstruktur ist die Standard-Einstellung des Spring-Security Moduls, welche die Autorisierungsgruppen (Authorities) widerspiegelt. Wie im Konzept (Kapitel 3.2.2.3) beschrieben, gibt es zwei verschiedene Benutzergruppen: Die Benutzergruppe `ROLE_USER` und `ROLE_ADMIN`. Erstere entsprechen den externen Benutzern, welche die Daten der Ersteller (zweite Gruppe) anreichern. Weiters wurde ein User mit dem Namen `anonymous` angelegt, welcher für anonyme Benutzer verwendet wird. Dieser wurde angelegt, weil auch anonyme Benutzer Bewertungen abgeben können und diese Bewertungen einem User zugeordnet werden müssen. Ansonsten würde ein Fremdschlüssel-Constraint verletzt werden. Nachdem jeder Benutzer einer Organisation zugeordnet werden muss, wurde die Organisation "SystemOrganisation" angelegt und dem User `anonymous` zugeordnet (Fremdschlüssel `organisation_id` in der Tabelle `user`). Weitere Details zu Spring-Security werden in Kapitel 4.3.4 erörtert.

## 4.3 Serverarchitektur

Auf der Serverseite wird Spring in der Version 2.5.6 als Application Framework eingesetzt[9]. Spring ist modular aufgebaut und stellt viele nützliche Funktionen zur Verfügung. Konkret kamen die Module `DAO` für den Datenzugriff sowie `Spring Security` zur Authentifikation zum Einsatz.

Das vorherrschende Entwurfsmuster, das in Spring zur Anwendung kommt, ist "Inversion of Control" (IoC), also "Steuerungsumkehrung". In einer herkömmlichen Anwendung sind die Objekte selbst für die Zuordnung der richtigen Services, die sie benötigen, verantwortlich. Spring übernimmt das als unabhängige dritte Partei und injiziert den Objekten die entsprechenden Abhängigkeiten. Die Objekte kennen nur die Schnittstellen, um die Services anzusprechen, und nicht die Services selbst. So wird eine lose Kopplung erreicht. Die Methodik des Injizierens der Abhängigkeiten wird auch als "Dependency Injection" bezeichnet (vgl. [Zeitner et al., 2008]).

In Abbildung 4.3 ist das Komponentendiagramm des Servers zu sehen.
`SpringCore` ist der Spring-IoC-Container, von dem alle anderen Komponenten abhängen. Dieser enthält und verwaltet den Lebenszyklus und die Konfiguration von Anwenundungsobjekten. Konkret ist die `BeanFactory` für die Dependency Injection verantwortlich. Sie fungiert als Fabrik und erzeugt neue Objekte (Beans), welche nicht

[9] Auch wenn Spring aktuell in der Version 3.0.0 erhältlich ist, zur Zeit der Entwicklung war diese noch in der Betaphase.

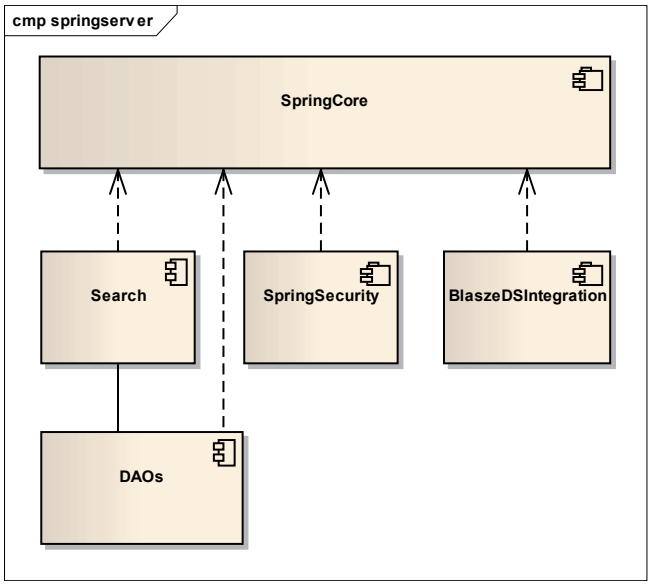



Abbildung 4.3: Komponentendiagramm des Servers

der Java-Bean-Spezifikation[10] entsprechen müssen. Der Spring Container inklusive der Beans wird über eine XML-Datei konfiguriert, in der die Beans miteinander verschaltet werden. Das `search`-Modul bildet die Geschäftslogik ab, die restlichen drei sind selbstsprechend. Im folgenden wird auf die einzelnen Komponenten näher eingegangen.

### 4.3.1 Geschäftslogik

Bevor auf die Geschäftslogik selbst eingegangen wird, werden die Business Objects beschrieben. Diese werden in den weiteren Abschnitten erwähnt und eine kurze Erklärung dieser ist für das weitere Verständnis von Vorteil.

Spring verwendet für die Datenhaltung per default sog. POJOs (Plain Old Java Object). Ein POJO ist ein simples Java Objekt, das im Gegenzug zu den komplexeren Enterprise Java Beans Verwendung in Spring fand. POJOs haben, ebenso wie Beans, Getter und Setter, und oft weitere Methoden zum Verändern des Verhaltens. Die Business Objects der OpenSearch Anwendung sind in Abbildung 4.4 zu sehen. Die fünf Klassen korrespondieren mit den fünf gleichnamigen Tabellen des Datenbankschemas und implementieren alle das `Serializeable` Interface, welches für die Serialisierung und Übertragung zu Flex-Clients notwendig ist. Dies wird in Abschnitt 4.3.3 im Detail behandelt. Die Texte der Erfolgsfaktoren werden in `SFText`-Objekten gespeichert. Diese besteht nur aus den Members `id` und `text`. Die von den Benutzern

[10] `http://java.sun.com/javase/technologies/desktop/javabeans/docs/spec.html`

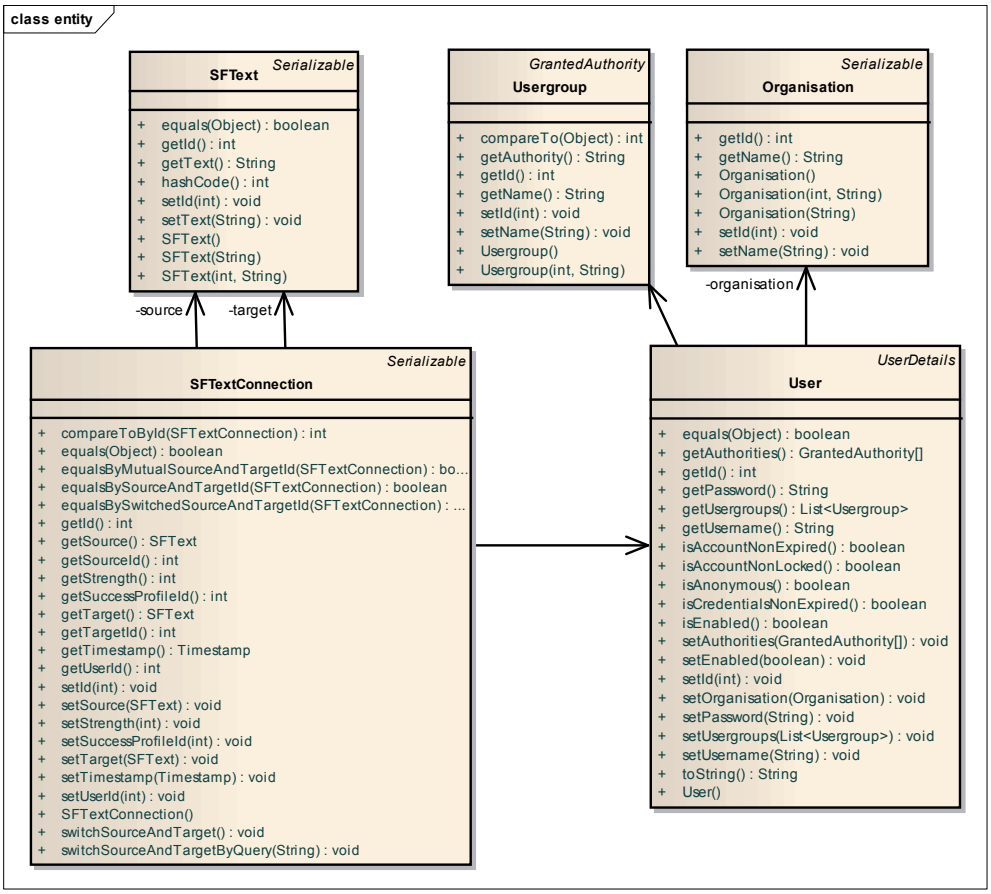



Abbildung 4.4: Business Objects

gespeicherten Verbindungen sind `SFTextConnection` Objekte. Diese bestehen, wie alle Datenobjekte, aus einer `id` sowie den Referenzen zu Texten, die zueinander in Beziehung stehen (`source`, `target`). Die drei Klassen rechts sind zwar keine Business Objects im eigentlichen Sinne, sondern für die Benutzerverwaltung notwendig.

#### 4.3.1.1 OpenSearchEngine

Das `search`-Modul beinhaltet die Geschäftslogik der Anwendung und erfüllt zwei Aufgaben: Die Erfolgsverbindungen suchen und die Bewertungen der Benutzer speichern. Diese Funktionalität ist über das Interface `OpenSearchEngine` definiert, welche in der Service-Klasse `OpenSearchEngineImpl` implementiert ist. Die Abbildung 4.5 zeigt das dazugehörige Klassendiagramm der `OpenSearchEngine`.
Die Implementation `OpenSearchEngineImpl` hat drei Referenzen auf die DAOs[11] `OpenSearchDAO`, `SFTextConnectionDAO` und `UserSFTextConnectionRatingDAO` sowie auf den `UserManager`, über welchen auf den Kontext des aktuellen Benutzers zugegriffen werden kann.

[11]Data Access Object

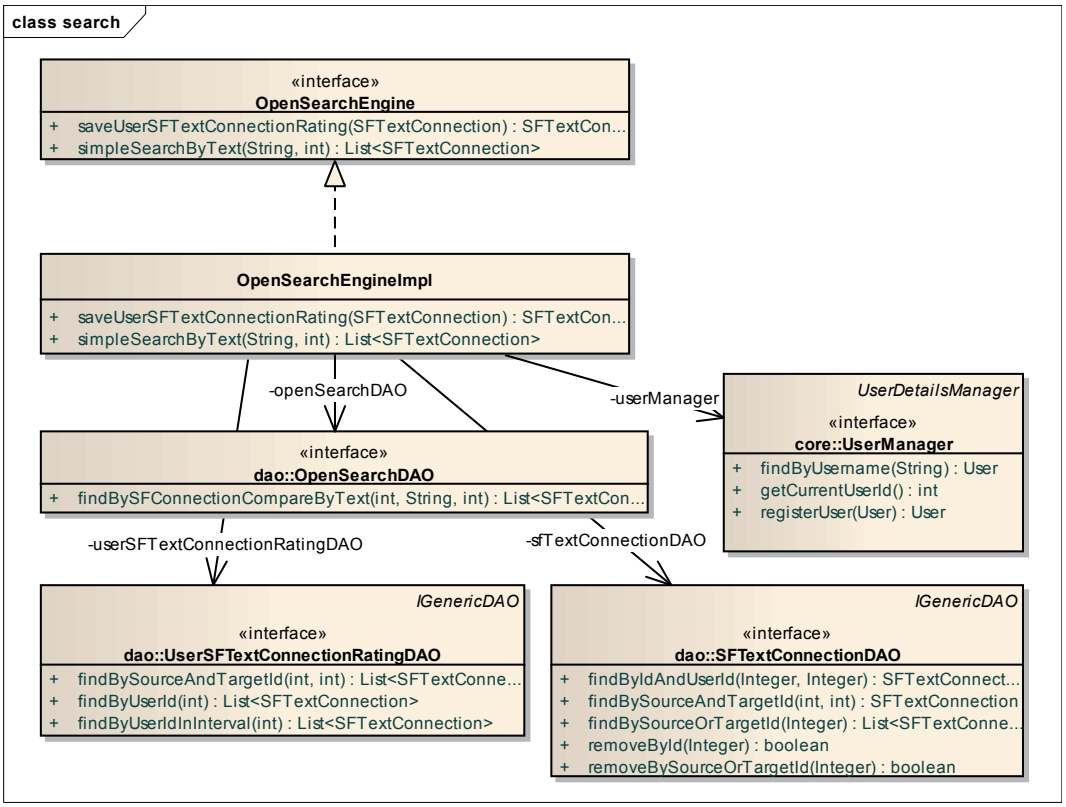



Abbildung 4.5: Klassendiagramm OpenSearchEngine

Alle vier sind Interfaces, um eine möglichst lose Kopplung zu schaffen.

#### 4.3.1.2 Suche

Über die Methode

```
List<SFTextConnection> simpleSearchByText(String query, int rowCount)
```

von `OpenSearchEngine` wird die Suche gestartet. Der Parameter `query` gibt den zu suchenden Text an und `rowCount` die Anzahl an Suchresultaten, die zurückgeliefert werden sollen.
Zuerst wird über den `UserManager` die `userId` des verbundenen Clients ermittelt. Es wird nun unterschieden, ob ein anonymer oder ein registrierter Benutzer eine Anfrage gesendet hat. Für anonyme Benutzer wird die Methode

```
findRandomBySFConnectionAndCompareByTextWithOutRatingCondition(
                        userId, query, rowCount);
```

von `OpenSearchDAO` aufgerufen, welche eine SQL-Abfrage an die Datenbank sendet, die folgendes tut: Die Texte der Quell- und Zielerfolgsfaktoren der existierenden Erfolgsprofile sind vom Typ `VARCHAR` und werden mit dem `query`-String per SQL-`LIKE`-Operator verglichen. Dies ergibt eine Ergebnismenge, von der mittels dem SQL-

Konstrukt `ORDER BY RAND() LIMIT 3` drei zufällige Datenreihen ausgewählt werden. Zusätzlich werden die von den Benutzern bisher abgegebenen Bewertungen in die Berechnung der Wirkungsstärke miteinbezogen. D.h., wenn Bewertungen zu dieser Verbindung bereits abgegeben wurden, so wird deren Mittelwert berechnet, und der Mittelwert zwischen Original-Stärke (von einem Erfolgsprofil-Ersteller) und dem Mittelwert der externen Bewertungen berechnet.
Für registrierte Benutzer wird nahezu das gleiche SQL-Statement abgesetzt, nur wird eine weitere Bedingung hinzugefügt: Es werden all jene Verbindungen nicht in die Ergebnismenge miteinbezogen, welche der Benutzer innerhalb der letzten 24 Stunden bewertet hat. Damit sollen dem Benutzer möglichst wenige gleiche Erfoglsverbindungen präsentiert werden, um die Wahrscheinlichkeit zu erhöhen, dass er andere Verbindungen bewertet. Werden mit dieser Abfrage keine Resultate gefunden, so wird das gleiche Statement wie für anonyme Benutzer abgesetzt, um dem Benutzer dennoch Ergebnisse zu liefern. Diese hat er zwar schon bewertet, allerdings ist das für den Benutzer motivationssteigernder, als keine Resultate zu erhalten.

Schlussendlich wird die Liste noch für die Übertragung zum Client aufbereitet. Dazu wird aus der Suchresultatsliste eine komplette wechselseitige Liste erstellt. Das bedeutet, dass für alle Verbindungen die wechselseitige Verbindung gesucht und direkt hinter die jeweilige Verbindung angefügt wird. Zur Erklärung folgt ein Bespiel: In einer Verbindung wirkt der Text A auf B, die wechselseitige Beziehung dazu lautet B auf A. Ist keine Verbindung vorhanden, so wird dennoch eine wechselseitige Verbindung eingefügt, deren Wirkung leer ist.

#### 4.3.1.3 Bewertung

Um Bewertungen von Usern speichern zu können, muss die zweite Methode, die `OpenSearchEngineImpl` zur Verfügung stellt, aufgerufen werden:

```
SFTextConnection saveUserSFTextConnectionRating(SFTextConnection conn)
```

Der Parameter `connection` ist die zu speichernde Textverbindung. Hierzu wird zuerst über den `UserManager` die `userId` der Verbindung gesetzt. Danach wird mittels `SFTextConnectionDAO` eine neue Verbindung angelegt, sofern diese nicht existiert. Die Bewertung wird danach über das `UserSFTextConnectionRatingDAO` in die Datenbank gespeichert und an den Client returniert.

### 4.3.2 Datenzugriff

Spring stellt mit dem DAO Modul bereits sämtlichen Code für Datenbankanbindungen sowie Templates für DAOs zur Verfügung. In dieser Webanwendung wurde die Datenbankverbindung über JDBC realisiert. Folgende Bean-Konfiguration reicht aus, um über eine Datenquelle eine Verbindung zur MySQL Datenbank in Spring herzustellen:

```
<bean id="dataSource"
  class="...datasource.DriverManagerDataSource">
  <property name="driverClassName" value="com.mysql.jdbc.Driver" />
  <property name="url"
            value="jdbc:mysql://localhost:3306/successprofiler" />
  <property name="username" value="xxxx" />
  <property name="password" value="xxxx" />
</bean>
```

Codebeispiel 4.1: Definition der JDBC-Datenquelle in Spring

Es wurden bereits die drei relevanten DAOs für `OpenSearchEngine` erwähnt. Zusätzlich existieren noch die DAOs `UserDAO`, `UsergroupDAO` und `OrganisationDAO` für die Persistierung in die Datenbank der Business Objects `User`, `Usergroup` und `Organisation`. Der Datenzugriff in Spring wird über Datenzugriffstemplates geregelt. Exemplarisch folgt das Klassendiagramm für das `UserSFTextConnectionRatingDAO`.
Auf linken Seite des Diagramms sind zwei proprietäre Interfaces definiert, auf der rechten Seite die Klassenhierarchie des Spring DAO-Templates. Spring stellt auch noch andere DAO-Templates zur Verfügung, im Rahmen der Entwicklung der Webanwendung wurden allerdings alle DAOs über die Klasse `NamedParameterJdbcDaoSupport` realisiert. Mit diesem ist es möglich, einem DAO die Namen der Parameter für SQL-Statements zu übergeben, anstatt nur die Indizes der Parameter. Dadurch ist der Code wartbarer, weil er flexibler gegenüber Änderungen in der Datenbank ist. Anhand von `UserSFTextConnectionRatingDAO` wird demonstriert, wie diese Bean in der Applikations-Konfigurationsdatei mit der `datasource` vom obigen Codebeispiel verschaltet werden muss, um auf die gewünschte Datenbank zuzugreifen. Dies zeigt das nachfolgende Codebeispiel:

```
<bean id="userSFTextConnectionRatingDAO"
  class="...dao.jdbc.UserSFTextConnectionRatingDAOImpl">
  <property name="dataSource" ref="dataSource" />
</bean>
```

Codebeispiel 4.2: Bean-Verschaltung von UserSFTextConnectionRatingDAO mit datasource

Auch alle anderen DAOs sind so mit der Datenquelle `datasource` verschaltet. Natürlich können bei Bedarf auch andere Datenbanken eingebunden werden. Es müssen nur die Parameter (`url`, `username`, `password`) richtig eingestellt werden und eine andere `id` vergeben werden. Und diese `id` muss beim Property `datasource` angegeben werden.

### 4.3.3 Remoting

BlazeDS ist der Java Server, der benötigt wird, um mit Flex-RPC Komponenten zu kommunizieren. BlazeDS hat eine Nachrichten-orientierte Archtitektur, wobei Nachrichten über Kanäle mit dem Client ausgetauscht werden. Die Kanäle werden ihrerseits zu Channel-Sets gruppiert. Z.B. gibt es einen HTTP-Kanal, der über das

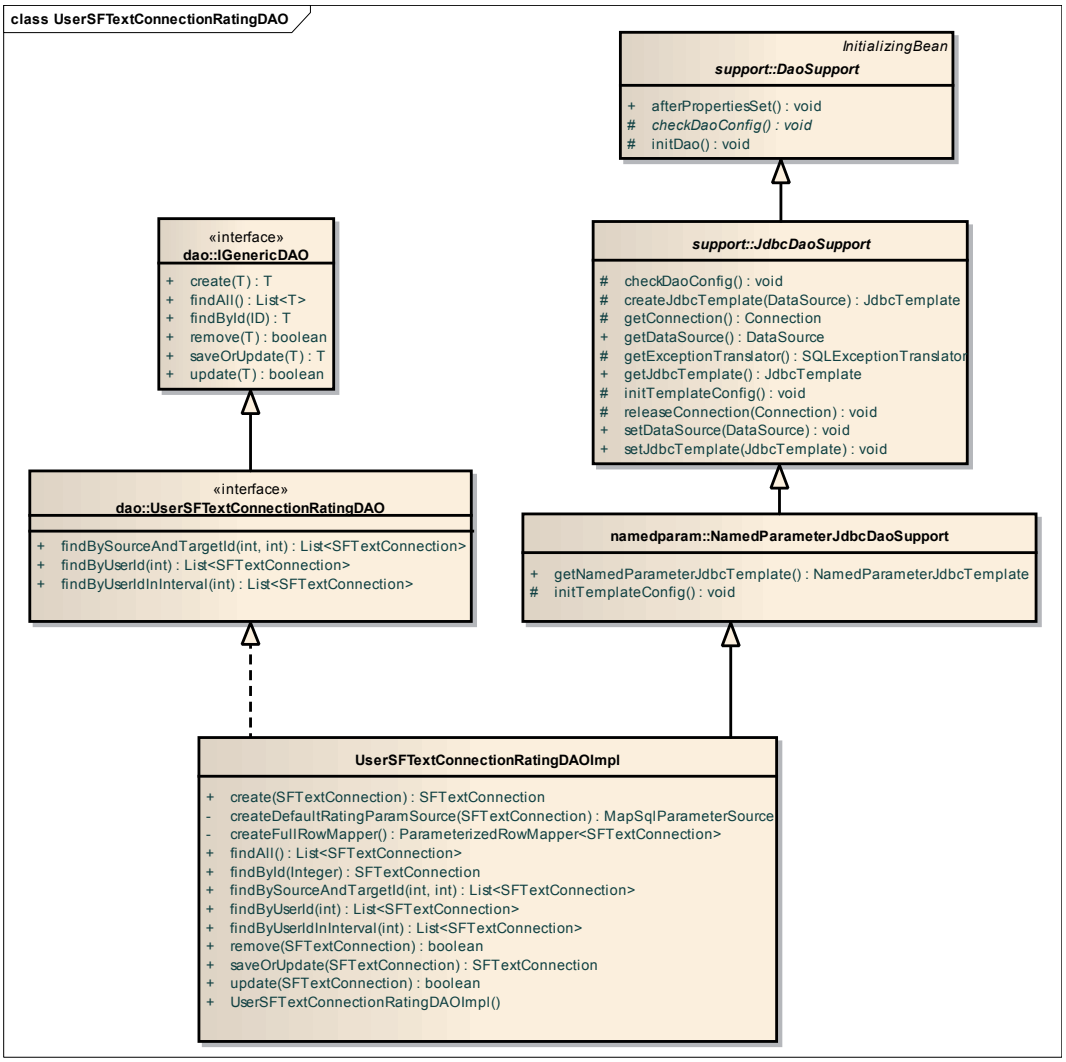



Abbildung 4.6: Klassendiagramm UserSFTextConnectionRatingDAOImpl

HTTP-Protokol kommuniziert und einen AMF-Kanal, welcher den Nachrichtenaustausch über AMF zulässt. Zweiterer wurde in der Implementierung verwendet, denn dadurch kann über das `Remote Object` von Flex-Clients auf den Server zugegriffen werden, der eine Java-zu-ActionScript Serialisierung durchführt. Somit muss z.B. kein XML mehr geparst werden, sondern es werden AchtionScript Objekte empfangen [Giametta, 2009].

Konkret ist das Java-Servlet (`MessageBroker`) für den Nachrichtenaustausch zuständig. Mit Hilfe von Spring BlazeDS Integration wird BlazeDS in Spring so integriert, dass der Spring Container auch BlazeDS verwaltet, ohne eine eigenständige BlazeDS Konfiguration außerhalb von Spring zu benötigen. Auf die genaue Konfiguration wird

an dieser Stelle verzichtet., es sei hier an die Referenz-Dokumentation[12] verwiesen.

Um den Zugriff auf Spring Beans von Flex über BlazeDS zu ermöglichen, reicht es aus, die erstellten Beans als `Remoting-Destination` in der Spring Konfiguration auszuzeichnnen. Exemplarisch wurde dies für die Bean `OpenSearchEngine` gemacht:

```
<flex:remoting-destination ref="openSearchEngine" />
```

Das Attribut `ref` gibt die Bean an, die von Flex-Clients erreichbar sein soll. So wurden auch alle anderen Beans, auf die ein Zugriff notwendig ist, ausgezeichnet.

### 4.3.4 Security

Für Sicherheitsfeatures wurde Spring Security verwendet. Dieses ist ein sehr mächtiges und zugleich flexibles Framework. BlazeDS stellt bereits grundsätzliche Security Features wie eine Authentifizierung bereit, aber die Spring BlazeDS Integration erweitert diese, sodass Spring Security in BlazeDS eingesetzt werden kann. Das zentrale Sicherheitsfeature, das benötigt wurde, ist die Authentifizierung.

Für die Authentifizierung benötigt Spring Security einen sog. `AuthenticationManager`. Folgendes Codebeispiel zeigt die Spring-Konfiguration des `AuthenticationProviders`:

```
<beans:bean id="userManager"
  class="...core.UserManagerImpl" />

<beans:bean id="passwordEncoder"
  class="...encoding.Md5PasswordEncoder" />

<beans:bean id="authenticationProvider"
  class="...dao.DaoAuthenticationProvider">
  <beans:property name="userDetailsService" ref="userManager" />

  <beans:property name="passwordEncoder" ref="passwordEncoder" />
  <beans:property name="saltSource" >
    <beans:bean
      class="...dao.salt.ReflectionSaltSource" >
      <beans:property name="userPropertyToUse" value="username" />
    </beans:bean>
  </beans:property>

  <custom-authentication-provider />
</beans:bean>
```

Codebeispiel 4.3: Spring-Konfiguration des AuthenticationProviders

[12] `http://static.springsource.org/spring-flex/docs/1.0.x/reference/html/index.html`

Je nachdem, ob man sich gegenüber einer Datenbank, einem LDAP-Server[13] oder CAS[14] authentifizieren möchte, stellt Spring verschiedene Authentication Provider zur Verfügung. Nachdem Benutzername und Passwort in der Datenbank gespeichert werden, wird ein sog. `DaoAuthenticationProvider` eingesetzt, welcher die Authentifizierung durchführt. Dieser `DaoAuthenticationProvider` benötigt ein DAO, das die Benutzerdaten aus der Datenbank abfragt. Dieses aufgrund der Übersichtlichkeit im Codebeispiel weggelassen.

Als erste Bean wurde der `UserManager` erstellt. Dieser implementiert das Interface `UserDetailsSerivice`, welches den Zugriff auf ein Benutzer-Objekt erlaubt, das im aktuellen Kontext der Authenfizierung steht. Durch die Angabe des `PasswordEncoder` wird festgelegt, dass das Passwort verschlüsselt gespeichert ist. In diesem Fall wird eine MD5 Verschlüsselung verwendet (`Md5PasswordEncoder`). Beim Abgleich mit dem übergegebenen Passwort, das vom Client zum Server im Klartext übertragen wird, wird es beim Authentifizierungsversuch verschlüsselt und mit dem Passwort aus der Datenbank (durch `DaoAuthenticationProvider`) verglichen. Natürlich muss es beim Registrieren verschlüsselt in die Datenbank gespeichert werden. Hierfür wird auch die Bean `PasswordEncoder` verwendet.

Für die Registrierung ist der `UserManager` verantwortlich, der eine entsprechende Methode bereitstellt. Dieser greift intern auf das `UserDAO` zurück, welches die Benutzerinformationen in die Datenbank schreibt. Ebenso erfolgt im Rahmen des Registrierungsprozesses nach dem Anlegen des Benutzers eine Zuordnung zu einer Benutzergruppe. Für die Persistierung in die Datenbank ist das `userGroupDAO` verantwortlich. An dieser Stelle wird nicht auf weitere Details eingegangen, da es den Rahmen der Arbeit sprengen würde. Im nächsten Kapitel wird die Umsetzung des Clients beschrieben.

## 4.4 Client

Als Client-Technologie wurde wie bereits erwähnt das Adobe Flex Framework und damit die Flash-Plattform gewählt. Für den Client wurde das MVC-Framework PureMVC[15] eingesetzt. Dieses trennt die Anwendung (wie alle MVC-Frameworks) in drei Schichten: Model (Datenmodell), View (Präsentationsschicht) und Controller (Programmsteuerung). Dadurch wird ein flexibler Programmentwurf erreicht, der spätere Änderungen oder Erweiterungen erleichtert sowie die Wiederverwendbarkeit der einzelnen Komponenten fördert und ermöglicht. Die drei Komponenten Model, View und Controller sind Singletons und werden als “Hauptakteure” bezeichnet. PureMVC stellt mit der Facade ein viertes Singleton zur Verfügung, über dessen

[13] Lightweight Directory Access Protocol
[14] Central Authentication Service
[15] `http://puremvc.org/`

Interface die Kommunikation zwischen den Hauptakteuren einfach gehalten wird. PureMVC setzt zur Kommunikation zwischen den Hauptakteuren und anderen Teilen des Systems ein eigenes Observer-Modell ein. Die Kommunikation erfolgt dabei über Notifications [Hall, 2009].

Die umgesetzte Client-Architektur ist in Abbildung 4.7 zu sehen. Das Model hält Referenzen zu Proxies, welche Daten speichern und den Zugriff auf lokale oder auch entfernte Daten bewerkstelligen. Proxies können Notifications senden, allerdings keine empfangen, da das Model sonst zu eng an View und Controller gekoppelt wäre. Die View hält Referenzen zu Mediators, welche die UI-Komponenten verwalten und Event-Listener hinzufügen. Der Status der Komponenten wird durch Mediators verändert, indem Notifications versendet und empfangen werden. Der Controller hält Mappings zu Command-Klassen, über welche die Proxies verändert und abgerufen werden. Die Controller ist somit die Verbindung von Model und View.

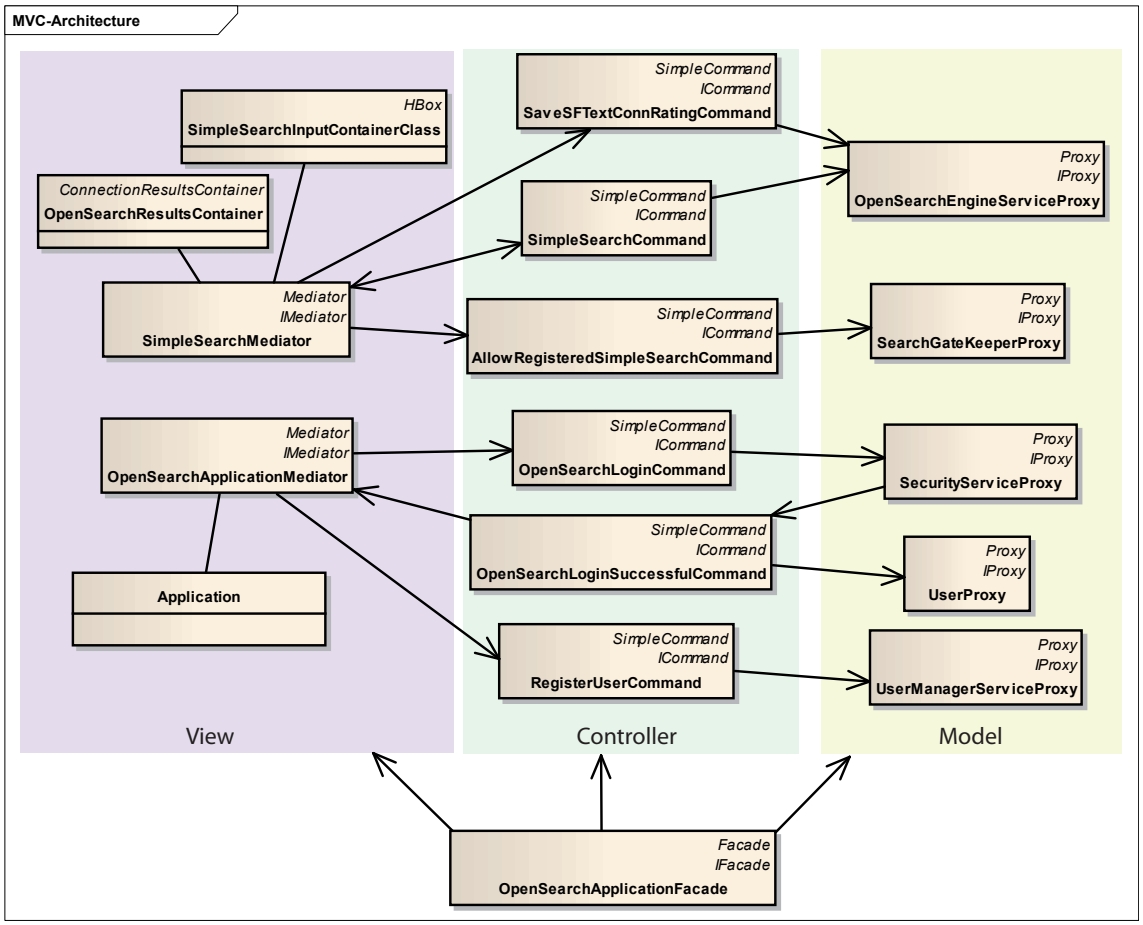



Abbildung 4.7: MVC-Architektur des Clients

Das Diagramm enthält die wichtigsten Komponenten. Die folgenden Kapitel enthal-

ten Details dazu.

### 4.4.1 Model

Das Model besteht es aus den Proxies, welche die Domain Objects verwalten, die im weiteren Sinne Pendants zu den Domain Objects in Java sind. Abbildung 4.8 zeigt die verwendeten Proxy-Klassen.

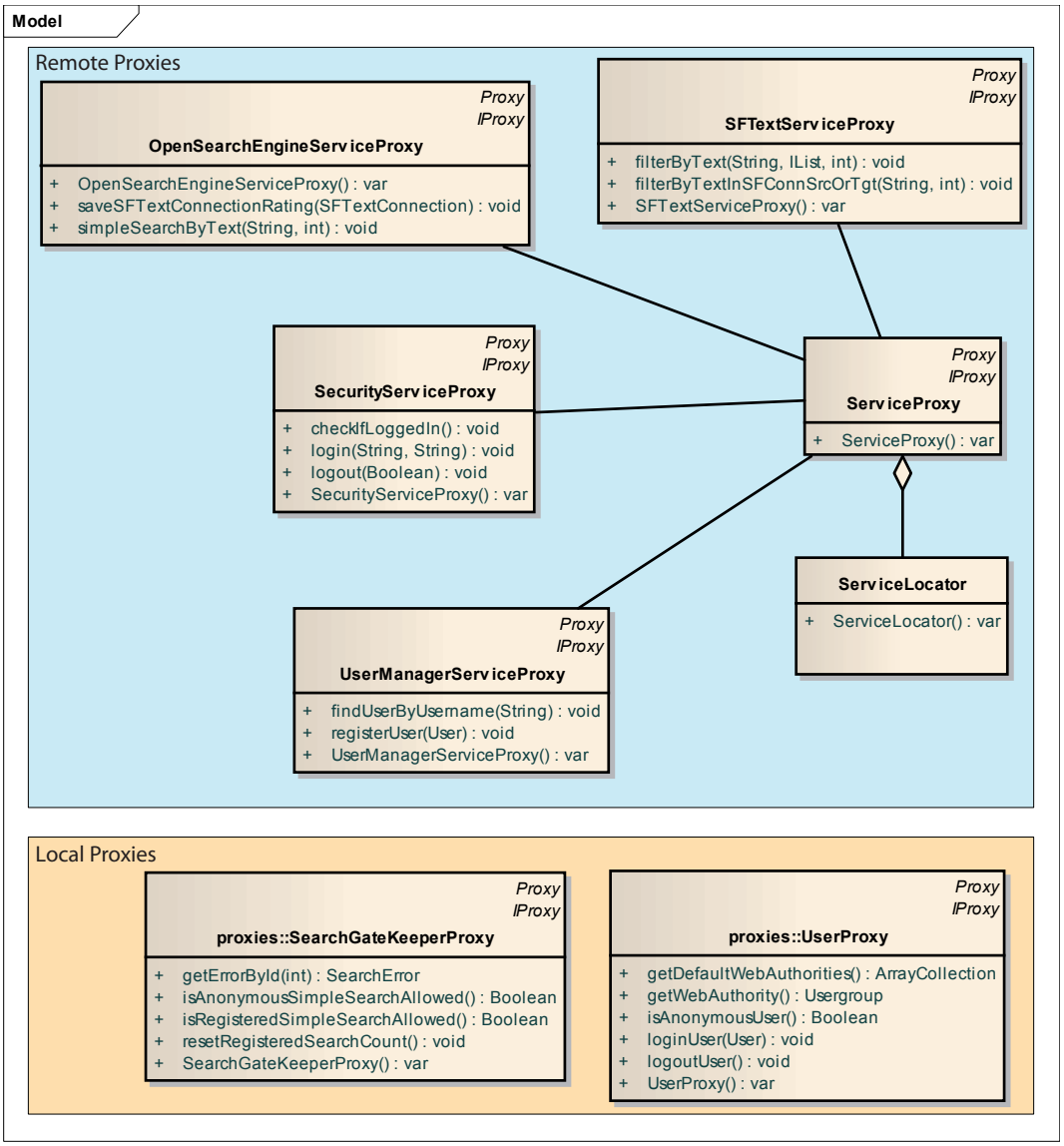



Abbildung 4.8: Klassendiagramm der Proxies (Model)

Die Proxies werden in Lokale und Remote Proxies unterschieden: Lokale Proxies verwalten lokale Daten, und Remote Proxies greifen auf die Java Services des Backends

zu.

#### 4.4.1.1 Lokale Proxies

`SearchGateKeeperProxy` hält nur zwei Zählvariablen. Je einen für Suchen von einem anonymen und von einem angemeldeten User. Bei jeder Suche wird einer der Zähler hinaufgesetzt (je nachdem ob ein User angemeldet ist oder das System als anonymer User verwendet).

Der `UserProxy` speichert den Status eines Users, damit auf die User-Daten wie z.B. den Namen zugegriffen werden kann.

#### 4.4.1.2 Remote Proxies

Die Remote Proxies ermöglichen den Zugriff auf die Backend Services mittels `RemoteObjects`. Wie in Kapitel 4.3.3 erwähnt, erfolgt die Datenübertragung über einen AMF-Kanal, um `RemoteObjects` nutzen zu können. Um den Kanal zu nutzen, muss zuerst ein ChannelSet definiert werden, wie das folgende Codebeispiel zeigt:

```
private var amfChannelSet_:ChannelSet = new ChannelSet();
amfChannelSet_.addChannel(ServerConfig.getChannel("my-amf"));
```

Codebeispiel 4.4: Definition des AMF-ChannelSet

In der zweiten Zeile wird der AMF-Kanal hinzugefügt. Eine Referenz dazu wird aus der `ServerConfig` geholt, welche ein Teil des Adobe Flex SDKs ist. Diese Konfiguration kann beim Kompilieren als Parameter mitgegeben werden, um sie nicht manuell eintragen zu müssen und flexibel gegenüber Änderungen am Server zu bleiben. Im nächsten Beispiel wird das `RemoteObject` für das `OpenSearchEngineService` erstellt:

```
private var openSearchEngineService_:RemoteObject;
openSearchEngineService_ = new RemoteObject();
openSearchEngineService_.destination = "openSearchEngine";
openSearchEngineService_.makeObjectsBindable = true;
openSearchEngineService_.showBusyCursor = true;
openSearchEngineService_.channelSet = amfChannelSet_;
```

Codebeispiel 4.5: Erstellung des RemoteObject für das OpenSearchEngineService

Der Parameter `destination` in Zeile zwei gibt die ID der Bean am Spring Server an, auf die zugegriffen werden soll. Diese wurde in der Spring-Konfiguration mittels `remoting-destination` ausgezeichnet (vgl. dazu Kapitel 4.3.3). In der letzten Zeile wird noch die Referenz auf das zuvor definierte ChannelSet übergeben. Nun kann durch folgenden Aufruf die Methode für die Suche nach Textverbindungen abgesetzt werden:

```
var token:AsyncToken = service_.simpleSearchByText(query,rowCount);
```

Die Parameter sind dieselben wie in der Spring Bean definiert. Die Kommunikation per `RemoteObject` verläuft im Gegensatz zu HTTP-Anfragen asynchron. D.h., das

Senden und Empfangen der Daten findet zeitlich versetzt statt. Dadurch wird der aufrufende Prozess (Flex-Client Applikation) nicht unterbrochen und kann weiterlaufen. Allerdings muss eine Callback-Funktion angegeben werden, die aufgerufen wird, wenn der Server antwortet. Dies geschieht über einen `AsyncResponder`, für den eine Funktion definiert werden muss, die aufgerufen wird, wenn eine Anfrage erfolgreich war und eine Funktion, die Fehler behandelt. Das folgende Codebeispiel zeigt den Responder inklusive der Methode, die aufgerufen werden soll, wenn der Aufruf von `simpleSearchByText` erfolgreich war:

```
token.addResponder( new AsyncResponder(
  function(data:Object, token:Object):void {
    var result:Object = (data as ResultEvent).result;
    var dataProvider:ArrayCollection = result as ArrayCollection;

    sendNotification(
      OpenSearchApplicationFacade.SIMPLE_SEARCH_RESULT,
      dataProvider );
  },
  defaultErrorHandler,
  null
));
```

Codebeispiel 4.6: AsyncResponder mit Result- und Error-Callback

In Zeile zwei ist die Inline-Definiton der Callback-Funktion zu sehen: Der Parameter `data` ist ein `ResultEvent`-Objekt, in dem das Resultat gespeichert ist. In Zeile vier wird das `result` in eine `ArrayCollection`[16] gecastet. Dieses ist ein vollwertiges ActionScript Objekt, das nun einsatzbereit ist. In diesem Beispiel ist das Result-Objekt eine Liste von `SFTextConnection`-Objekten. Damit die Serialisierung zwischen Java und ActionScript funktioniert, muss die Klasse mit `RemoteClass` annotiert werden, wie folgendes Beispiel zeigt (Zeile 1):

```
[RemoteClass(alias="at.opensearch.entity.SFTextConnection")]
public class SFTextConnection {
  private var id_:int;
  public function SFTextConnection() {}

  public function get id():int {
    return id_;
  }

  public function set id(id:int):void {
    id_ = id;
  }
  // restliche Parameter nicht angezeigt
  ...
```

Codebeispiel 4.7: Definition eines Value Objects zur Serialisierung zwischen ActionScript und Java

[16] Diese ist eine Liste und mit der `ArrayList` in Java gleichzusetzen

Der Parameter `alias` gibt den Fully Qualified Class Name der Java Klasse an. Zusätzlich müssen für sämtliche in Java definierten public Getter-Methoden in der ActionScript Klasse ein Getter und Setter definiert werden.

Die Service-Proxies in der Anwendung erhalten ihre `RemoteObjects` über den `ServiceProxy`, der die Schnittstelle zum `ServiceLocator` ist. Der `ServiceLocator` erzeugt die `RemoteObjects`. Der `OpenSearchEngineServiceProxy` ist die Schnittstelle zur `OpenSearchEngine`, `SecurityServiceProxy` stellt Login und Logout-Methoden zur Verfügung und der `UserManagerServiceProxy` ermöglicht den Zugriff auf den `UserManager`, um neue Benutzer zu registrieren.

### 4.4.2 Controller

Die `ApplicationFacade` kapselt den Zugriff auf den Controller. Dieser wird durch die Commands repräsentiert, welche beim Starten der Facade in der Methode `initialzeController()` registriert und auf die Notifications gemappt werden. Folgendes Beispiel zeigt das Registrieren für das Login-Command beim Controller:

```
registerCommand( LOGIN, OpenSearchLoginCommand );
```

Dabei ist der erste Parameter `LOGIN` der Name der Notification und der zweite die Klasse, welche das Command beinhaltet. Commands trennen die View vom Model, greifen auf das Model zu und verändern dieses. Ein Command hat eine einzige Methode, das es überschreibt:

```
override public function execute(notification:INotification):void
```

Wenn eine Notification gesendet wurde, werden die Mappings der Notifications zu den Commands überprüft. Existiert ein Mapping, so wird eine Instanz des gemappten Commands erstellt und dessen `execute`-Methode aufgerufen. Im obigen Beispiel wird ein neues `OpenSearchLoginCommand` erstellt, wenn die Notification `LOGIN` gesendet wird. Als Parameter wird eine `INotification` übergeben, welche einen Namen und einen `Body` als Members hat. Der Body dient zum Kapseln von Parametern bzw. Daten, die für das Verändern des Models gebraucht werden.

Die wichtigsten Notifications und gleichnamigen, gemappten Commands sind:

- `LOGIN`: Wird vom `SimpleSearchMediator` gesendet, wenn auf den Login-Button geklickt wurde. Als Parameter wurden Username und Passwort übergeben. Das Command ruft die `login`-Methode des `SecurityServiceProxy` auf, welcher den User am Server anmeldet.

- `LOGIN_SUCCESSFUL`: Bei erfolgreicher Anmeldung mit Username und Passwort am Server wird die Notification `LOGIN_SUCCESSFUL` gesendet. Das gleichnamige Command loggt daraufhin den Benutzer über den `UserProxy` auch am Client ein, sodass auch der Client über den Zustand des Users informiert ist. So

kannn überprüft werden, ob ein anonymer oder ein angemeldeter User mit dem Anwendung interagiert.

- `REGISTER_USER`: Wird aufgerufen, wenn ein neuer Benutzer registriert werden soll. Es werden dabei Username, Passwort und die Wiederholung des Passworts als Parameter übergeben. Sind die Eingaben korrekt, so wird über den `UserManagerServiceProxy` die `registerUser`-Methode aufgerufen, welcher einen neuen Benutzer am Server anlegt.

- `TRY_SIMPLE_SEARCH`: Wird gesendet, bevor die Suchabfrage an den Server gestartet wird. Es wird überprüft, ob eine Suche erlaubt ist oder nicht. So wird mittels `UserProxy` festgestellt, ob der User anonym oder angemeldet ist. Mit Hilfe des `SearchGateKeeperProxy` wird ermittelt, ob bereits mehr als drei Suchen abgesetzt wurden. Ist dies der Fall, so werden je nach Anmeldestatus des Benutzers (anonym oder angemeldet) entsprechende Fehlermeldungen angezeigt. Ist die Suche gestattet, so wird die `SIMPLE_SEARCH`-Notification gesendet.

- `SIMPLE_SEARCH`: Über den `OpenSearchEngineServiceProxy` wird die Suche mittels der Methode `simpleSearchByText` abgesetzt. Der Server erstellt die Suchresultate, und wenn diese beim Client ankommen, wird die `SIMPLE_SEARCH_RESULT`-Notification gesendet. Dazu gibt es kein gemapptes Command, da sich der `SimpleSearchMediator` für diese Notification registriert hat. Dieser stellt die Suchresultate dar. Weitere Details dazu im nächsten Abschnitt unter 4.4.3.2.

- `SAVE_USER_SF_TEXT_CONNECTION`: Wird aufgerufen, wenn ein User eine Bewertung abgegeben hat. Es wird mittels `OpenSearchEngineServiceProxy` die `SFTextConnection` an den Server gesendet, welcher die Bewertung in die Datenbank schreibt.

- `SAVE_USER_SF_TEXT_CONNECTION_RESULT`: Diese sendet nur die Notification `ALLOW_REGISTERED_SIMPLE_SEARCH`.

- `ALLOW_REGISTERED_SIMPLE_SEARCH`: Es wird über den `SearchGateKeeperProxy` der Counter für angemeldete Benutzer zurückgesetzt, wodurch in weiterer Folge wieder drei Suchen abgesetzt werden dürfen.

### 4.4.3 View

Die View besteht zum einen aus der Benutzerschnittstelle und zum anderen aus den Mediators, welche diese verwaltet. Die Mediators erhalten beim Erstellen eine Referenz auf eine UI-Komponente, die sie verwalten sollen. Es ist auch möglich, dass ein Mediator mehrere UI-Komponenten verwaltet. In der Anwendung gibt es zwei Mediators: `OpenSearchApplicationMediator` und `SimpleSearchMediator`.

#### 4.4.3.1 OpenSearchApplicationMediator

Der `OpenSearchApplicationMediator` ist für das Login- und Registrierungs-Fenster sowie für die Menüleiste zuständig. Er legt beim Erstellen das Login- und Registrierungs-Fenster an und fügt entsprechende Event-Listener für die Butttons in der Menüleiste sowie das Login-Fenster hinzu. Wie bereits erwähnt, können Mediators Notifications empfangen. Damit sie nicht alle empfangen, müssen sie Interesse an den gewünschten Notifications bekunden. Dies wird mit Hilfe der Methode `listNotificationInterests` realisiert, welche jeder Mediator vererbt und überschreiben muss bzw. soll. Folgendes Codebeispiel zeigt die Notifications für den `OpenSearchApplicationMediator`:

```
override public function listNotificationInterests():Array
{
  return [ApplicationFacade.LOGIN_SUCCESSFUL,
          ApplicationFacade.LOGOUT_SUCCESSFUL,
          ApplicationFacade.REGISTER_USER_RESULT]
}
```

Codebeispiel 4.8: Die Notifications für den OpenSearchApplicationMediator

Diese Funktion returniert ein Array mit den Notification-Namen. Damit wird der Mediator nur bei diesen benachrichtigt und kann dementsprechend reagieren. Wie die Notifications bearbeitet werden, wird in der Methode `handleNotification` realisiert:

```
override public function handleNotification(
                notification:INotification):void
{
  switch ( notification.getName() )
  {
    case ApplicationFacade.LOGIN_SUCCESSFUL:
      successfulLoginCallback();
      break;

    case ApplicationFacade.LOGOUT_SUCCESSFUL:
      succesfulLogoutCallback();
      break;

    case ApplicationFacade.REGISTER_USER_RESULT:
      registerUserResultCallback();
      break;
}
```

Codebeispiel 4.9: Notification-Handler

Das `successfulLoginCallback()` wird aufgerufen, wenn eine Anmeldung am Server erfolgreich war: Die Benutzereingaben des Loginfensters werden gelöscht und das Fenster versteckt. Der Login-Button wird in Logout-Button umbenannt und dessen Event-Listener wird auf `logout` gesetzt. Der Registrieren-Button wird versteckt und der Text des Status-Label in der Menüleiste zeigt den Usernamen des gerade eingeloggten Users an. Das `successfulLogoutCallback()` ist das Pendant zu

`successfulLoginCallback()` und sorgt dafür, dass das GUI wieder in den alten Zustand zurückkehrt.

#### 4.4.3.2 SimpleSearchMediator

Der `SimpleSearchMediator` ist für die Anwendungslogik, also die Suche und die Bewertungen, zuständig. Er fügt die Events für das Absenden der Suchanfrage hinzu, und stellt die Suchresultate in Listenform dar. Er belegt die Rating-Balken mit Events, um in weiterer Folge die Bewertungen absetzen zu können.

Das `simpleSearchResultsCallback` wird aufgerufen, wenn die Suchergebnisse einer soeben abgesetzten Suche ankommen. Es wird über die Liste mit `SFTextConnection`s iteriert und für je zwei Ergebnisse eine Ergebnisreihe mit zwei Rating-Balken (sog. `ToggleButtonBar`s) im `SearchResultsContainer` erstellt. Dieser wurde so konfiguriert, dass beim Hinzufügen einer Ergebnisreihe das Event für das Klicken auf einen Button des Rating-Balken gesetzt wird, welches die Notification `SAVE_USER_SF_TEXT_CONNECTION` sendet.

Eine Suchergebnisreihe ist vom Typ der Klasse `SFTextConnectionsComboRow`, dessen Klassendiagramm in Abbildung 4.9 zu sehen ist.

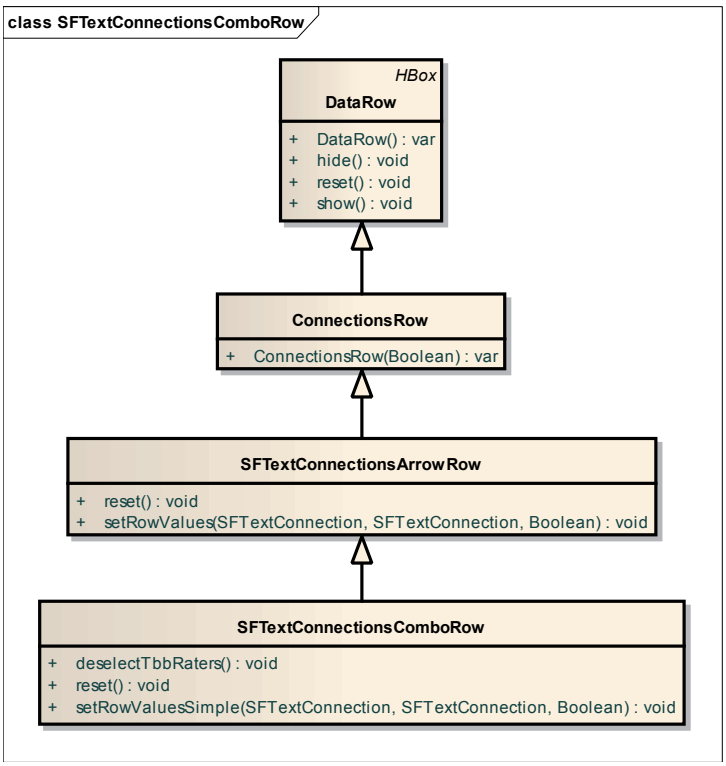



Abbildung 4.9: Klassendiagramm SFTextConnectionsComboRow

Die Basisklasse ist `DataRow`, welche von der Flex-Klasse `HBox` erbt. Dies ist eine Box, dessen Inhalt horizontal angeordnet wird. `ConnectionsRow` ist die Klasse, die zwei Labels und einen Container in der Mitte der beiden (zum Anzeigen der Pfeile sowie der Rating-Balken) beinhaltet. `SFTextConnectionsArrowRow` fügt dem Container ein `Canvas`-Container hinzu, in das die Pfeile gezeichnet werden, welche die Wirkung der beiden Texte bzw. Erfolgsfaktoren darstellen. Für die Datenhaltung werden zwei `SFTextConnection`-Member Variablen angelegt. `SFTextConnectionsComboRow` erweitert die vorhin erwähnte Klasse um die Rating-Balken. Diese werden erst angezeigt, wenn der Benutzer über einen der Pfeile mit der Maus fährt. Die Pfeile werden hingegen versteckt. So kann der Benutzer die Bewertungen abgeben.

#### 4.4.3.3 Graphisches User Interface

Flex stellt eine Fülle an Layout-Containern zur Verfügung. Die Anwendung ist in drei Container gegliedert: Die Menüleiste, den Such-Container mit Suchtextfeld und Suchbutton sowie dem `SearchResultsContainer`, welcher die Suchresultate beinhaltet.

Abbildung 4.10 zeigt die Suchresultate eines anonymen Benutzers, welcher nach "strateg" gesucht hat:

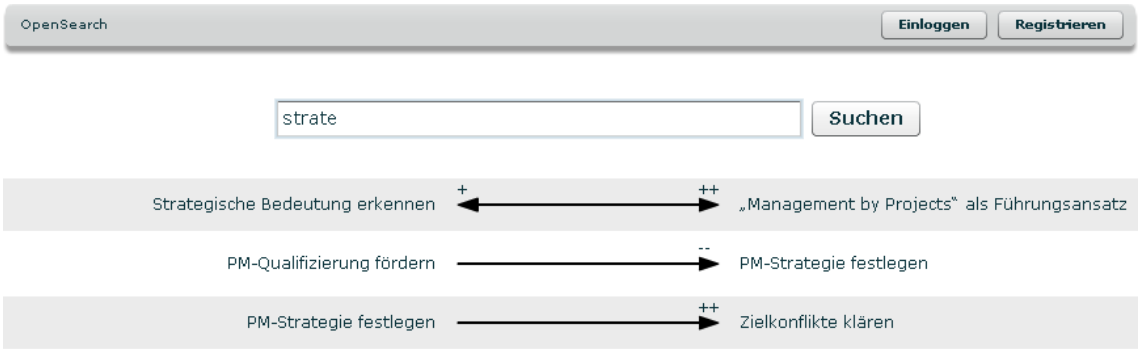



Abbildung 4.10: GUI: Suchresultate für anonymen User

Abbildung 4.11 zeigt das Login-Fenster auf der nächsten Seite.

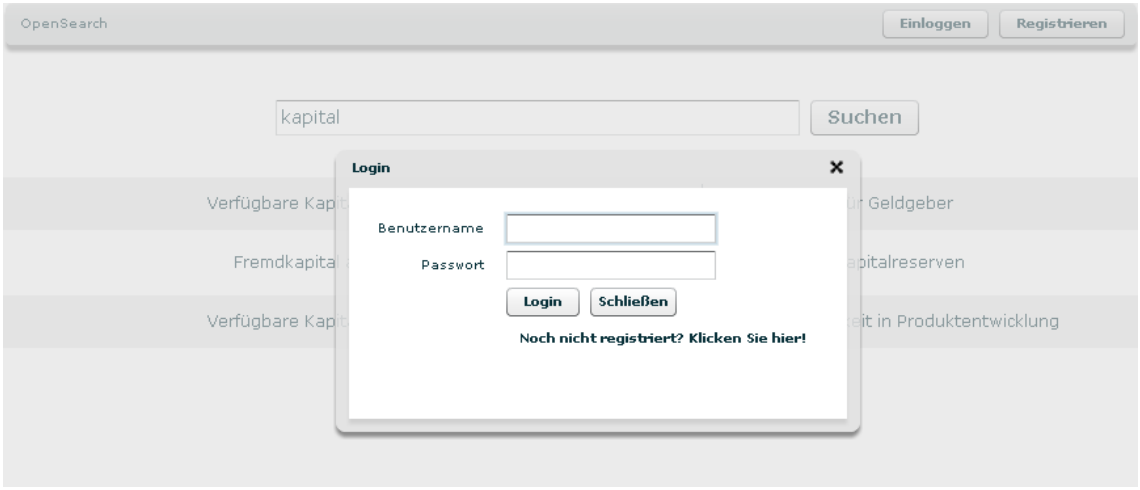



Abbildung 4.11: GUI: Login-Fenster

In Abbildung 4.12 ist der Bewertungsbalken zu sehen, der erscheint, wenn ein User über die Wirkungs-Pfeile fährt. Auf der rechten Seite in der Menüleiste steht der Statustext, dass der User mit dem Namen "test" eingeloggt ist. Der "Registrieren"-Button wurde versteckt, und der "Einloggen-Button wurde in "Ausloggen" umbenannt.

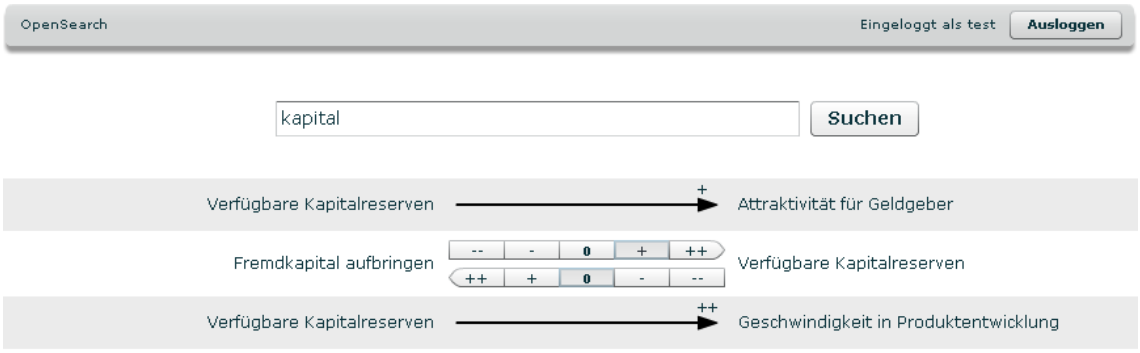



Abbildung 4.12: GUI: Bewertung eines angemeldeten Benutzers

# Kapitel 5

# Evaluierung

Die Evaluierung der Anwendung ist eine qualitative Bewertung und erfolgte im Rahmen einer Pilotnutzung des Systems durch drei Experten aus dem Umfeld der SUCCON.

## 5.1 Ablauf

Die Pilotnutzungen erfolgten jeweils in folgenden Stufen:

1. Kurze Darstellung der Hintergründe und Überlegungen zum OpenSearch-System ("Die OpenSearch als Tool zur Unterstützung beim Querdenken und Innovieren")
2. Eigenständige Testnutzung der Webanwendung durch die Pilotnutzer
3. Abschließende Einzelgespräche bzw. -befragungen mit den Pilotnutzern zu den Kernthemen

Abschließend erfolgte die Zusammenführung aller Evaluierungsergebnisse.

## 5.2 Testaufbau

Getestet wurde clientseitig auf einem handelsüblichen Notebook. Das Backend wurde auf einem virtuellen Server des Know-Center bzw. Instituts für Wissensmanagement betrieben. Es folgt die detaillierte Hard- und Softwarekonfiguration der Testumgebung.

Die Konfiguration des Clients ist in Tabelle 5.1 zu sehen. Die Tabelle darunter zeigt die Konfiguration des Servers. Dieser wurde als einer von zehn virtuellen Servern auf einer Dell-Server-Hardware betrieben.

| **Hardware** | |
|---|---|
| Notebook | HP |
| Prozesor | Intel Core 2 Duo 2GHz |
| Arbeitsspeicher | 2GB RAM |
| **Software** | |
| Betriebssystem | Windows XP SP3 |
| Browser | Internet Explorer 8 |
| Flash Plugin | Version 10.0.42.34 |

Tabelle 5.1: Hard- und Softwarekonfiguration des Clients

| **Hardware (physikalisch)** | |
|---|---|
| Server | Dell Power Edge 2950 |
| Prozesor | Intel Xeon E5420 2,5GHz (8 Kerne) |
| Arbeitsspeicher | 16GB RAM |
| **Hardware (virtuell)** | |
| Prozesor | 1GHz |
| Arbeitsspeicher | 1GB RAM |
| Anmerkung | eine von insgesamt zehn virtuellen Server Instanzen |
| **Software** | |
| Betriebssystem | Windows Server 2003 Standard Edition 32Bit |
| Virtualisierungssoftware | VMWare ESXi |
| Webserver | Apache Tomcat 6.0.18 |
| Datenbankserver | MySQL 5.0.27 |

Tabelle 5.2: Hard- und Softwarekonfiguration des Servers

Für den Test wuren 20 Erfolgsprofile erstellt. Diese bestanden in Summe aus ca. 380 Erfolgsfaktoren und 500 Erfolgsverbindungen.

## 5.3 Ergebnisse

Es folgt eine Auflistung der Ergebnisse kategorisiert nach Thematik. Zuerst werden such-, danach bewertungsspezifische Ergebnisse präsentiert. Zuletzt werden Anmerkungen das GUI betreffend geschildert. Zu Beginn jedes Themenpunktes wird jeweils die Frage angeführt, welche den Pilotnutzern zur jeweiligen Thematik im abschließenden Interview gestellt wurde.

### 5.3.1 Relevanz der Suchresultate

*"Haben Sie relevante Suchresultate erhalten?"*

Aus Sicht der Pilotuser besitzen die Suchresultate großteils Relevanz. In Verbindung mit jenen Suchresultaten, die eine eher geringere oder keine Relevanz besitzen, wurde rasch klar, dass die Relevanz bzw. Qualität der Suchresultate primär von den vorhandenen Daten bzw. von den in den Ursprungsdaten definierten Wirkungsbeziehungen abhängt und nicht vom Funktionieren der Suchfunktionalität.
Für eine noch effektivere Suche wäre es hilfreich, könnte man entscheiden, welche Rolle bzw. Position der jeweilige Suchbegriff in den Suchergebnissen einnehmen soll. So könnte man gezielt danach suchen, worauf (auf welche Faktoren) ein bestimmter Erfolgsfaktor wirkt oder aber von welchen anderen Faktoren ein bestimmter Erfolgsfaktor beeinflusst wird (im ersten Fall wäre der Suchbegriff "Wirkungsquelle", im zweiten Fall "Wirkungsziel"). Ebenso würde aus Sicht der Pilotnutzer die Möglichkeit zur Einschränkung der Datenquellen für die Suche deren Effektivität steigern. Mit Datenquellen sind in diesem Zusammenhang die Erfolgsprofile gemeint, aus denen die Erfolgsfaktoren extrahiert werden und als Basis für die Erstellung der Suchresultate dienen. Nachdem diese nicht einsehbar sein sollen, müsste dazu ein Kategoriensystem per vorgegebener, ausgewählter Kategorien oder durch frei wählbare Tags erstellt werden.

### 5.3.2 Breite Streuung der Suchresultate

*"Ist es störend, wenn verschiedene Ergebnisse bei gleicher Suche geliefert werden?"*

Dies wurde durch die Nutzer nicht als störend empfunden. Da das System beim Querdenken unterstützen soll, sind möglichst breite bzw. unterschiedliche Ergebnisse bei gleichen, jedoch im zeitlichen Ablauf hintereinander abgesetzten Suchen besonders hilfreich. Je mehr mögliche Erfolgsfaktorenverbindungen (d. h. je mehr unterschiedliche Suchergebnisse aufgezeigt werden, umso mehr Hinweise und Ansatzpunkte für erfolgssteigernde und innovative Lösungen werden dem "Sucher" präsentiert.

Aus Sicht der Pilotnutzer wäre es besonders hilfreich, würde im Zusammenhang mit dem OpenSearch-System eine Funktion "Persönliche Favoriten" (ähnlich den Bookmarks im Browser) angeboten. Diese Funktion sollte es ermöglichen, besonders relevante Suchergebnisse (Erfolgsfaktoren und deren Wirkungsbeziehungen) in einem jederzeit wieder abrufbaren "Favoritenordner" abzulegen.

### 5.3.3 Geschwindigkeit der Suche

*"Wie empfinden Sie die Geschwindigkeit der Suche, was die Antwortzeit und die Darstellung der Suchergebnisse betrifft?"*

Bei den meisten Pilotsuchen war die Geschwindigkeit bzw. Response Time der Suche sehr zufriedenstellend. Interessant war es zu erkennen, dass an unterschiedlichen Tageszeiten leicht unterschiedliche Responsezeiten gegeben waren, was aller Voraussicht nach auf die unterschiedliche Netzwerk- bzw. Serverauslastung zurückzuführen war.

### 5.3.4 Alternative Bewertungen

*"Welche anderen Formen der Bewertung können Sie sich vorstellen?"*

Die Pilotnutzer schlugen folgende drei alternativen Bewertungsformen vor:

- Bewertung mithilfe von Schiebereglern: So müsste die in manchen Fällen schwierige Entscheidung zwischen schwach positiver (bzw. negativer) Wirkungsbeziehung (+) und stark positiver (bzw. negativer) Wirkungsbeziehung (++) nicht so "eindeutig" getroffen werden.

- Bewertung durch Zustimmung/Nicht-Zustimmung (Thumbs Up/Down): Diese Form der Bewertung würde eventuell eine raschere Bewertung der Wirkungsbeziehungen erlauben, da man lediglich *einer* bereits definierten Ausprägung der Wirkungsbeziehung zustimmen (oder nicht zustimmen) müsste und sich nicht Gedanken über alle möglichen Beziehungsausprägungen (derzeit fünf) machen müsste.

- Bewertung im Sinne von Gewichtung: Dies ist keine alternative, sondern eine zusätzliche Form der Bewertung. Zusätzlich zur Definition der Wirkungsbeziehungen könnte es hilfreich sein, auch die Bedeutung/Gewichtung einzelner Faktorenverbindungen abzufragen bzw. anzugeben. So würde bei den Suchergebnissen dann nicht nur die Ausprägung der Wirkungsbeziehungen angezeigt werden, sondern auch deren Bedeutung für den Erfolg aus Sicht der Nutzer.

### 5.3.5 Freiweillige Bewertung

*"Würden Sie freiwillig bewerten, auch wenn Sie Suchresultate erhalten würden, ohne "zwischendurch" bewerten zu müssen?"*

Die Meinungen der Pilotnutzer hierzu waren ursprünglich recht unterschiedlich. Eindeutig war aber in den Einzelgesprächen zu erkennen, dass einer freiwilligen Bewertung umso eher zugestimmt wurde, je mehr die Pilotnutzer über die Hintergründe

und Überlegungen zum OpenSearch-System erfuhren. Allen Pilotnutzern wurde dabei klar, dass die Qualität/das Nutzenpotenzial des Systems beinahe ausschließlich von der Anzahl und Qualität an Einzelbewertungen abhängig ist. Auch wurde von den Pilotnutzern aufgezeigt, dass die Qualität/Mächtigkeit des Systems steigt, je mehr OpenSearch-Nutzer zu verzeichnen sind und vor allem je mehr davon als "Langzeitnutzer" einzustufen sind. Denn nur derjenige, der auch künftig vor hat, die Anwendung selbst zu nutzen, wird aus Sicht der Pilotnutzer ausreichend motiviert sein, Bewertungen abzugeben und damit die Qualität und Attraktivität des Systems laufend zu steigern.

### 5.3.6 Verbesserungspotenziale am GUI

> *"Soll eine Rückmeldung erscheinen, dass die Bewertung gespeichert wurde, wenn man auf die Bewertungsskala klickt?"*
>
> *"Haben Sie noch weitere Wünsche bzw. Anmerkungen zur Verbesserung des User Interfaces?"*

Für die Pilotnutzer war nicht gänzlich klar, ob und wann ihre eigene Bewertung gespeichert wurde. Aus deren Sicht wäre es hilfreich, könnte man mit einem "Senden-Button" seine Bewertung jeweils abschließen und würde daraufhin eine Rückmeldung wie z.B. "Ihre Bewertung wurde gespeichert, vielen Dank! Sie können nun eine erneute Suchanfrage absetzen." erhalten. Auch würde direkt nach Klicken auf einen Button der Bewertungsskala eine dementsprechende Meldung nützlich sein, sofern sie nicht zu "aufdringlich" ist und automatisch wieder verschwindet (ohne ein Eingreifen des Benutzers).

In der aktuellen Form bereitete manchen Pilotnutzern die Bewertung der Wirkungsbeziehungen Schwierigkeiten. Aufgrund der geringen (Zeilen-)Abstände zwischen den "Bewertungspfeilen" fiel es den Nutzern einerseits schwer, im Rahmen der Bewertung den Blick in der jeweils betrachteten/behandelten Zeile zu halten. Andererseits war es ihnen zum Teil schlecht möglich, die von ihnen bereits getätigten Bewertungen auf einen Blick zu erkennen. Im Zusammenhang mit der Bewertung wäre es aus Sicht der Pilotnutzer hilfreich, könnte man den Zeilenabstand vergrößern. Auch wäre es hilfreich, würden die von den Nutzern selbst bewerteten Wirkungsbeziehungen nach deren Bewertung optisch hervorgehoben.

Am Login-Fenster wurden keine Verbesserungspotenziale erkannt. Ebenso wurden zur Ladezeit keine Verbesserungspotenziale definiert, da diese zur Zeit des Test testumgebungsbedingt schwankte.

# Kapitel 6

# Zusammenfassung und Ausblick

## 6.1 Zusammenfassung

Im Rahmen der Masterarbeit wurden existierende Plattformen zur Diskussion von strukturierten Informationen beschrieben. Im Rahmen des dieser Masterarbeit zugrundeliegenden Auftrags der SUCCON Schachner & Partner KG an die TU Graz wurde eine Rich Internet Application zur Diskussion von bestehenden, strukturierten Informationen aus einem speziellen Anwendungsfall konzipiert und entwickelt. Ziel war es, die vorhandenen Informationen durch Dritte diskutieren bzw. anreichern zu lassen, um eine andere Sichtweise zu erhalten. In der Einleitung wurde der Anwendungsfall im Zuge der Motivation näher beschrieben. Ergänzend wurden zwei Szenarien vorgestellt, wie dieser Use Case in einer Anwendung umgesetzt werden kann.

Der theoretische Hintergrund wurde in Kapitel 2 betrachtet. Nach der Definition grundlegender Begriffe wurden drei Diskussionsplattformen von strukturierten Informationen im Web 2.0 beschrieben: Blogs, Wikis und Foren. Für jede Plattform wurden neben typischen Merkmalen und charakteristischer Funktionen das Potenzial der Diskussion i.A. erörtert. Jede Plattform wurde in den Kontext des Anwendungsfalles gebracht und so ihre Nutzungsmöglichkeiten zur Diskussion der vorliegenden strukturierten Informationen ermittelt. Es stellte sich heraus, dass keine der drei Plattformen für den Anwendungsfall geeignet ist. Dies hat zwei Gründe: Erstens müssen die bestehenden, strukturierten Informationen in (unstrukturierte) textuelle Form überführt werden, um eine Diskussion über die genannten Plattformen zu ermöglichen. Zweitens sind auch die Diskussionsbeiträge der Benutzer, welche die Daten anreichern sollten, unstrukturiert. Um die Informationen aus den Diskussionbeiträgen zu nutzen bzw. zu verwerten, müssten diese mittels Text Minining analysiert und extrahiert werden. Allerdings ist die erreichte Genauigkeit gering. Somit wurde ein anderer Lösungsweg aufgezeigt: Die Informationen müssen über eine Plattform so aufbereitet werden, dass die Diskussion in strukturierter Art und Weise ermöglicht wird. Als Umsetzungsplattform wurde eine Rich Internet Application

(RIA gewählt.

In Kapitel 3 wurden RIAs eingehend besprochen. Dabei wurde die geschichtliche Entwicklung ebenso beleuchtet wie Charakteristiken, Technologien und Architekturvarianten von RIAs. Im Abschnitt 3.2 wurde das Umsetzungskonzept für den Anwendungsfall dargelegt. Neben der Beschreibung der Anforderungen wurde deren geplante Umsetzung ausführlich geschildert. In Kapitel 4 wurde die Implementierung beschrieben. Zu Beginn wurde die RIA-Architektur im Überblick erklärt, sowie die eingesetzten Technologien und Gründe für deren Wahl erörtert. Das enwtickelte Datenbankschema wurde ebenso beschrieben wie das Server-Backend. Mit der Beschreibung des Frontends wurde das Kapitel abgeschlossen. Zu guter letzt wurde eine Evaluierung durchgeführt, die in Kapitel 5 beschrieben wurde. Sie erfolgte im Rahmen der Pilotnutzung des Systems durch drei Experten aus dem Umfeld der SUCCON. Diese bestand im Wesentlichen aus einer Testnutzung mit abschließenden Einzelbefragungen, deren Resultate schließlich zusammengeführt wurden. Daraus ließen sich zahlreiche Verbesserungsvorschläge ableiten, die im folgenden Abschnitt kurz beschrieben werden.

## 6.2 Zukünftige Arbeit

Es werden hier Verbesserungen am System besprochen, die im Rahmen der Evaluierung erfasst wurden, jedoch im Laufe der Arbeit nicht behandelt und umgesetzt werden konnten. Diese betreffen vor allem das User Interface, mehr Funktionalitäten sowie dadurch hervorgerufene Änderungen am Backend.

Was die Suche betrifft, ist es sinnvoll, den Usern erweiterte Sucheinstellungen anzubieten. Dazu zählt, dass man die Position des Suchbegriffs einstellen kann. Damit ist gemeint, dass der User auswählen kann, ob der Suchbegriff nur in Quell- oder Zielerfolgsfaktoren der Erfolgsverbindungen enthalten sein soll. Zur Zeit wird keine Unterscheidung getroffen, es werden Quelle als auch Ziel durchsucht.
Ein weiteres interessantes Feature wäre die Beschränkung der Datenquellen, in welchen die Erfolgsverbindungen gesucht werden sollen. Die Datenquellen sind die einzelnen Erfolgsprofile. Aufgrund der Geheimhaltung ist es jedoch unmöglich, User auswählen zu lassen, welche Erfolgsprofile durchsucht werden sollen. Deshalb müsste bereits in den Prozess des Erstellens von Erfolgsprofilen eingegriffen werden, und die Erfolgsprofile mit Schlagworten versehen oder kategorisiert werden. So bliebe die Anonymität gewahrt und eine spezifischere Suche wäre möglich.
Auch ist geplant, Erfolgsverbindungen als Favoriten ablegen zu können. Diese sollen für registrierte Benutzer abrufbar sein.

Auch die Usability, welche die Bewertungen betrifft, soll verbessert werden. Ein einfach zu implementierendes Feature ist das Anzeigen einer Benachrichtigung, wenn

auf einen Button der Bewertungsskala gedrückt wurde und die Bewertung erfolgreich gespeichert wurde. Bei kleinem Aufwand wird somit großer Nutzen durch GUI-Feedback erzielt. Zusätzlich scheint auch eine optische Hervorhebung der Verbindungen hilfreich, die von einem Benutzer zuvor bewertet wurden.

In der Evaluierung wurden verschiedene alternative Bewertungsmethoden erhoben (vgl. 5.3.4). Nach Rücksprache mit der SUCCON scheint es sinnvoll, in Zukunft auch weitere (einfachere) Bewertungsmöglichkeiten zu entwickelt und einzusetzen. So hätte z.B. eine Bewertung mittels Thumbs Up/Down den Vorteil, dass ein User einer bestehenden Verbindung zustimmt oder eben nicht der Meinung ist, dass zwischen den beiden Faktoren eine Beziehung steht.

Zur Zeit können die Beschränkungen des Systems einfach umgangen werden. Nach drei Suchen muss ein Benutzer eine Bewertung abgeben, durch ein Neuladen der Anwendung könnte der Benutzer dies jedoch einfach umgehen. Das gleiche gilt für registrierte Benutzer: Ein registrierter User kann drei freie Suchen absetzen. Meldet er sich nun ab und gleich danach wieder mit Benutzername und Passwort an, so braucht er nicht zu bewerten. Um so einen Missbrauch des Systems zu verhindern, könnte einerseits die IP-Adresse des Benutzers mitgespeichert werden, um ihn über eine Session hinweg zu identifizieren. Andererseits könnte auch durch die Verwendung von SharedObjects (wie ein Cookie in HMTL) eine Identifikation der Nutzer weiter dazu beitragen, dass das System nicht ausgenutzt wird.

# Literaturverzeichnis


[Abiteboul, 1997] Abiteboul, S. (1997). Querying semi-structured data. In Afrati, F. N. and Kolaitis, P. G., Herausgeber, *ICDT*, Band 1186 von*Lecture Notes in Computer Science*, Seiten 1–18. Springer.

[Abiteboul et al., 2000] Abiteboul, S., Buneman, P., and Suciu, D. (2000). *Data on the Web: From Relations to Semistructured Data and XML*. Morgan-Kaufmann, San Francisco.

[Adobe, 2009] Adobe (2009). Das flex-framework. Website. URL: `http://www.adobe.com/de/products/flex/features/flex_framework/` [Aufruf 19.11.2009].

[Adobe, 2010] Adobe, S. I. (2010). Flash player penetration. Website. URL: `http://www.adobe.com/products/player_census/flashplayer/` [Aufruf 16.01.2010].

[Alby, 2007] Alby, T. (2007). *Web 2.0. Konzepte, Anwendungen, Technologien*. Hanser.

[Allaire, 2002] Allaire, J. (2002). Macromedia flash mx-a next-generation rich client. Technical Report. URL: `http://www.adobe.com/devnet/flash/whitepapers/richclient.pdf`.

[Anderson, 2004] Anderson, C. (2004). The long tail. Website. URL: `http://www.wired.com/wired/archive/12.10/tail.html` [Aufruf 04.08.2009].

[Back et al., 2008] Back, A., Gronau, N., and Tochtermann, K., Herausgeber (2008). *Web 2.0 in der Unternehmenspraxis: Grundlagen, Fallstudien und Trends zum Einsatz von Social Software*. Oldenbourg.

[Bansal, 2009] Bansal, A. (2009). Disqus, intensedebate and js-kit analysis. Website. URL: `http://www.knowliz.com/2009/06/disqus-intensedebate-js-kit-analysis.html` [Aufruf 21.09.2009].

[Bartel, 2007] Bartel, R. (2007). *BLOGS für alle - das Weblog-Kompendium: Edition CHIP*. Smartbooks, 1 Auflage.

[Bendel, 2006] Bendel, O. (2006). *Das 1x1 der Wikis und Weblogs*, Band 3. wissensmanagement - Das Magazin für Führungskräfte.

[Berners-Lee and Hendler, 2001] Berners-Lee, T. and Hendler, J. (2001). The semantic web. a new form of web content that is meaningful to computers will unleash a revolution of new possibilities. *Scientific American Magazine*, Seiten 34–43.

[Bettel, 2008] Bettel, S. (2008). Warum web 2.0? oder: Was vom web2.0 wirklich bleiben wird. In Blumauer, A. and Pellegrini, T., Herausgeber, *Social Semantic Web. Web 2.0 - Was nun?*, Seiten 23–41. Springer, Berlin Heidelberg.

[Blood, 2002] Blood, R. (2002). *The Weblog Handbook: Practical Advice on Creating and Maintaining Your Blog*. Perseus Books, Cambridge MA.

[Bozzon et al., 2006] Bozzon, A., Comai, S., Fraternali, P., and Carughi, G. T. (2006). Conceptual modeling and code generation for rich internet applications. In *ICWE '06: Proceedings of the 6th international conference on Web engineering*, Seiten 353–360, New York, NY, USA. ACM.

[Burg, 2003] Burg, T. N. (2003). Weblogdefinition für wissensökologien. Website. URL: `http://randgaenge.net/texts/wissensoekologie/` [Aufruf 11.09.2009].

[C2.com, 2009a] C2.com (2009a). Why wiki works. Website. URL: `http://c2.com/cgi/wiki?WhyWikiWorks` [Aufruf 24.09.2009].

[C2.com, 2009b] C2.com (2009b). Wiki engines. Website. URL: `http://c2.com/cgi/wiki?WikiEngines` [Aufruf 24.09.2009].

[C2.com, 2009c] C2.com (2009c). Wiki history. Website. URL: `http://c2.com/cgi/wiki?WikiHistory` [Aufruf 24.09.2009].

[Cailloux, 2009] Cailloux, P. (2009). Echo feature-list. Website. URL: `http://wiki.js-kit.com/Feature-List` [Aufruf 23.09.2009].

[Connolly and Begg, 2005] Connolly, T. M. and Begg, C. E. (2005). *Database systems: a practical approach to design, implementation, and managemen*. Pearson Education.

[de Moor and Efimova, 2004] de Moor, A. and Efimova, L. (2004). An argumentation analysis of weblog conversations.

[Disqus, 2009] Disqus (2009). Social media reactions: Connect the conversation across the web. Website. URL: `http://blog.disqus.net/2009/04/02/social-media-reactions/` [Aufruf 23.09.2009].

[Domenig, 2005] Domenig, M. (2005). Rich internet applications and ajax - selecting the best product. Website. URL: `http://www.javalobby.org/articles/ajax-ria-overview/` [Aufruf 19.11.2009].

[Ebersbach et al., 2008a] Ebersbach, A., Glaser, M., Heigl, R., and Warta, A. (2008a). *Wiki: Kooperation im Web*. Springer, Berlin, Heidelberg, 2 Auflage.

[Ebersbach et al., 2008b] Ebersbach, A., Krimmel, K., and Warta, A. (2008b). *Auswahl und Aussage von Kenngrößen innerbetrieblicher Wiki-Arbeit*, Seiten 131–155. Vieweg+Teubner, Wiesbaden.

[Ebner, 2008] Ebner, M. (2008). *Internetforen: Verwenden- einrichten- betreiben*. BoD - Books on Demand.

[Ehrenstein, 2009] Ehrenstein, C. (2009). *Adobe Air - Grundlagen, Praxis, Referenz*. Galileo Design.

[Eichorn, 2006] Eichorn, J. (2006). *Understanding AJAX: Using JavaScript to Create Rich Internet Applications*. Prentice Hall.

[Farkas, 2007] Farkas, M. G. (2007). *Social Software in Libraries: Building Collaboration, Communication, and Community Online*. Information Today Inc.

[Farrell and Nezlek, 2007] Farrell, J. and Nezlek, G. S. (2007). Rich internet applications the next stage of application development. In *29th Int. Conference on Information Technology Interfaces (ITI 2007)*, Seiten 413–418, Allendale, MI, USA.

[ForumMatrix, 2009] ForumMatrix (2009). Forummatrix - compare them all. Website. URL: `http://www.forummatrix.org/` [Aufruf 12.10.2009].

[Gams and Mitterdorfer, 2009] Gams, E. and Mitterdorfer, D. (2009). Semantische content management systeme. In Blumauer, A. and Pellegrini, T., Herausgeber, *Social Semantic Web. Web 2.0 - Was nun?*, Seiten 207–226. Springer, Heidelberg. Inhaltsverzeichnis: http://swbplus.bsz-bw.de/bsz266277306inh.htm.

[Garrett, 2005] Garrett, J. J. (2005). Ajax: A new approach to web applications. Website. URL: `http://www.adaptivepath.com/ideas/essays/archives/000385.php` [Aufruf 25.07.2009].

[Giametta, 2009] Giametta, C. (2009). *Pro Flex on Spring*. Apress, Berkely, CA, USA.

[Glebe, 2008] Glebe, D. (2008). *Börse verstehen: Die globale Finanzkrise*. BoD Books on Demand.

[Godin, 2006] Godin, S. (2006). *Small Is the New Big: and 183 Other Riffs, Rants, and Remarkable Business Ideas*. Portfolio.

[Google, 2009] Google (2009). Gears - getting started. Website. URL: `http://code.google.com/intl/de-AT/apis/gears/design.html` [Aufruf 19.11.2009].

[Groß and Hülsbusch, 2004] Groß, M. and Hülsbusch, W. (2004). *Weblogs und Wikis - eine neue Medienrevolution?*, Band 8. wissensmanagement - Das Magazin für Führungskräfte.

[Gumbrecht, 2004] Gumbrecht, M. (2004). Blogs as "protected space". In *WWW 2004 Workshop on the Weblogging Ecosystem: Aggregation, Analysis and Dynamics*. URL: `http://www.blogpulse.com/papers/www2004gumbrecht.pdf`.

[Hall, 2009] Hall, C. (2009). Puremvc - implementierung, idiome und optimale anwendung. PDF. URL: `http://puremvc.org/pages/docs/current/PureMVC_Implementierung_Idiome_und_Optimale_Anwendung.pdf` [Aufruf 22.01.2010].

[Harman and Koohang, 2005] Harman, K. and Koohang, A. (2005). Discussion board: a learning object. 1:67–77.

[Heuer and Priebe, 2000] Heuer, A. and Priebe, D. (2000). Integrating a query language for structured and semi-structured data and ir techniques. In *DEXA Workshop*, Seiten 703–707. IEEE Computer Society.

[Hickson and Hyatt, 2009] Hickson, I. and Hyatt, D. (2009). Html5 - a vocabulary and associated apis for html and xhtml. Website. URL: `http://dev.w3.org/html5/spec/Overview.html` [Aufruf 17.11.2009].

[Holdener, 2008] Holdener, A. T. (2008). *Ajax: the definitive guide*. O'Reilly.

[ieblog, 2009] ieblog (2009). An early look at ie9 for developers. Website. URL: `http://blogs.msdn.com/ie/archive/2009/11/18/an-early-look-at-ie9-for-developers.aspx` [Aufruf 19.11.2009].

[Informationweek, 2006] Informationweek (2006). A brief history of web 2.0. Website. URL: `http://www.informationweek.com/1113/IDweb20_timeline.jhtml` [Aufruf 16.07.2009].

[Jangro, 2008] Jangro, S. (2008). Evaluating blog comment systems. Website. URL: `http://www.jangro.com/blogging/evaluating-blog-comment-systems/` [Aufruf 21.09.2009].

[JavaFX, 2009] JavaFX (2009). Javafx frequently asked questions. Website. URL: `http://www.javafx.com/faq/` [Aufruf 19.11.2009].

[John et al., 2005] John, M., Schmidt, S., and Decker, B. (2005). Community-management in unternehmen mit wiki- und weblogtechnologien. In Meißner, K; Engelein, M., Herausgeber, *Virtuelle Organisation und Neue Medien 2005. Workshop GeNeMe 2005 Gemeinschaften in Neuen Medien*, Seiten 105–119.

[Kaiser, 2008] Kaiser, R. (2008). *Bibliotheken im Web 2.0 Zeitalter*. B.I.T.online - innovativ ; 20. Dinges und Frick, Wiesbaden.

[Kane, 2009] Kane, C. (2009). Web browser javascript benchmark. Website. URL: `http://celtickane.com/labs/web-browser-javascript-benchmark` [Aufruf 16.11.2009].

[Kemper and Eickler, 2006] Kemper, A. and Eickler, A. (2006). *Datenbanksysteme*. Oldenbourg, 6., aktualisierte und erw. aufl Auflage.

[Koch and Richter, 2008] Koch, M. and Richter, A. (2008). *Enterprise 2.0 - Planung, Einführung und erfolgreicher Einsatz von Social Software in Unternehmen*. Oldenbourg, München.

[Koenig, 2009] Koenig, M. (2009). Friendfeed comment retrieval, gravatar enhancement, optional profanity filter, and email notification update. Website. URL: `http://blog.intensedebate.com/2008/06/18/friendfeed-comment-retrieval-gravatar-enhancement\-optional-profanity-filter-email-notification-update/` [Aufruf 23.09.2009].

[Kranz and Merz, 2005] Kranz, M. and Merz, M. (2005). Wikis in der softwareentwicklung. URL: `http://www.forumbb.de/files/SoftwareforumBB_WikiInSE_Condat.pdf`.

[Krempl, 2005] Krempl, S. (2005). Adobe übernimmt macromedia für 3,4 milliarden dollar. Website. URL: `http://www.heise.de/newsticker/meldung/Adobe-uebernimmt-Macromedia-fuer-3-4-Milliarden-Dollar-153865.html`.

[Krötzsch et al., 2007] Krötzsch, M., Schaffert, S., and Vrandecic, D. (2007). Reasoning in semantic wikis. In *Reasoning Web Summer School 2007*, Seiten 310–329.

[Lange, 2007] Lange, C. (2007). Web 2.0 zum mitmachen - die beliebtesten anwendungen. URL: `ftp://ftp.oreilly.de/pub/katalog/web20_broschuere.pdf`.

[Leuf and Cunningham, 2001] Leuf, B. and Cunningham, W. (2001). *The Wiki Way: Quick Collaboration on the Web*. Addison-Wesley Longman, Amsterdam, pap/cdr Auflage.

[Lindström, 2009] Lindström, J. (2009). Carakan. Website. URL: `http://my.opera.com/core/blog/2009/02/04/carakan` [Aufruf 16.11.2009].

[LKD, 2009] LKD (2009). Wikipedia:diskussionsseiten. Website. URL: `http://de.wikipedia.org/wiki/Wikipedia:Diskussionsseiten` [Aufruf 04.10.2009].

[Marx, 2009] Marx, D. (2009). Java ee and flex, part 1: A compelling combination. Website. URL: `http://www.javaworld.com/javaworld/jw-01-2009/jw-01-javaee-flex-1.html?page=2` [Aufruf 17.01.2010].

[Matsak, 2009] Matsak, A. (2009). Social comments. Website. URL: `http://wiki.developers.facebook.com/index.php/Social_Comments` [Aufruf 23.09.2009].

[McCullagh and Broache, 2007] McCullagh, D. and Broache, A. (2007). Blogs turn 10 – who's the father? Website. URL: `http://news.cnet.com/2100-1025_3-6168681.html` [Aufruf 10.09.2009].

[Microsoft, 2009] Microsoft (2009). Silverlight 3 - neue funktionen für rich-media websites und rich-interenet anwendungen. Website. URL: `http://www.microsoft.com/germany/net/silverlight/newfeatures.aspx` [Aufruf 19.11.2009].

[Microsoft, 2010] Microsoft, C. (2010). Microsoft silverlight release history. Website. URL: `http://www.microsoft.com/downloads/details.aspx?FamilyID=1B7A3205-B5F8-4E20-BF42-792DE5923454&displaylang=en` [Aufruf 16.01.2010].

[Möller, 2003] Möller, E. (2003). Das wiki-prinzip: Tanz der gehirne teil 1. Website. URL: `http://www.heise.de/tp/r4/artikel/14/14736/1.html` [Aufruf 24.09.2009].

[Möller, 2006] Möller, E. (2006). *Die heimliche Medienrevolution: Wie Weblogs, Wikis und freie Software die Welt verändern*. TelepolisMagazin der Netzkultur. Heise, Hannover, 2., erw. und aktualisierte aufl. Auflage.

[Münz, 2007] Münz, S. (2007). Foren und boards. Website. URL: `http://aktuell.de.selfhtml.org/artikel/gedanken/foren-boards/` [Aufruf 08.10.2009].

[Münz, 2009] Münz, S. (2009). *Webseiten professionell erstellen: Programmierung, Design und Administration*. Pearson Education.

[Moschovitis, 1999] Moschovitis, C. J. P. (1999). *History of the Internet: a chronology, 1843 to the present*. ABC-CLIO.

[Mozilla, 2009a] Mozilla (2009a). Firefox web browser - under the hood. Website. URL: `http://www.mozilla.com/en-US/firefox/performance/` [Aufruf 16.11.2009].

[Mozilla, 2009b] Mozilla (2009b). Moziall xul tutorial: Introduction. Website. URL: `https://developer.mozilla.org/en/XUL_Tutorial/Introduction` [Aufruf 19.11.2009].

[Mullenweg, 2009] Mullenweg, M. (2009). 6 steps to kill your community. Website. URL: `http://ma.tt/2009/08/kill-your-community/` [Aufruf 24.09.2009].

[Murugesan, 2007] Murugesan, S. (2007). Understanding web 2.0. *IT Professional*, 9(4):34–41.

[Musser and O'Reilly, 2006] Musser, J. and O'Reilly, T. (2006). *Web 2.0 Principles and Best Practices*. O'Reilly.

[Nardi et al., 2004] Nardi, B., Schiano, D., Gumbrecht, M., and Swartz, L. (2004). Why we blog. *Commun. ACM*, 47(12):41–46.

[Noda and Helwig, 2005] Noda, T. and Helwig, S. (2005). Rich internet applications - technical comparison and case studies of ajax, flash, and java based ria. Technical Report. URL: `http://www.uwebc.org/opinionpapers/docs/RIA.pdf`.

[Nokia, 2007] Nokia (2007). Nokia to acquire navteq. Website. URL: `http://www.nokia.com/press/press-releases/showpressrelease?newsid=1157198` [Aufruf 22.07.2009].

[Orchard et al., 2009] Orchard, L. M., Pehlivanian, A., Koon, S., and Jones, H. (2009). *Professional JavaScript Frameworks: Prototype,YUI, ExtJS, Dojo and MooTools*. Wrox.

[O'Reilly, 2005] O'Reilly, T. (2005). What is web 2.0? design patterns and business models for the next generation of software. Website. URL: `http://oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html` [Aufruf 10.07.2009].

[O'Reilly, 2006] O'Reilly, T. (2006). Web 2.0 compact definition: Trying again. Website. URL: `http://radar.oreilly.com/archives/2006/12/web-20-compact.html` [Aufruf 10.07.2009].

[Pawlowitz, 2001] Pawlowitz, N. (2001). *Kunden gewinnen und binden mit Online-Communitys: so profitieren Sie von Foren, Chats, Newsgroups und Newslettern*. Campus Verlag.

[Peters and Stock, 2007] Peters, I. and Stock, W. G. (2007). *Web 2.0 im Unternehmen*, Band 4. wissensmanagement - Das Magazin für Führungskräfte.

[Phillips, 2007] Phillips, C. (2007). Spanners for the virtual ego: an evaluation of facilities for online communities on the world wide web.

[phpBB, 2009] phpBB (2009). phpbb - creating communities worldwide. Website. URL: `http://www.phpbb.com/` [Aufruf 12.10.2009].

[Reed, 1999] Reed, D. P. (1999). That sneaky exponential - beyond metcalfe's law to the power of community building. Website. URL: `http://www.reed.com/gfn/docs/reedslaw.html` [Aufruf 17.07.2009].

[Research, 2006] Research, Z. (2006). Statistics on ebay web services usage. Website. URL: `http://blogs.zdnet.com/ITFacts/?p=10326` [Aufruf 24.07.2009].

[Runkehl et al., 1998] Runkehl, J., Schlobinski, P., and Siever, T. (1998). *Sprache und Kommunikation im Internet: Überblick und Analysen*. Vandenhoeck und Ruprecht.

[Saade and Huang, 2009] Saade, R. and Huang, Q. (2009). Meaningful learning in discussion forums: Towards discourse analysis. In *Proceedings of InSITE 2009: Issues in Informing Science and Information Technology*.

[Sauers, 2006] Sauers, M. P. (2006). *Blogging and RSS: a librarian's guide*. Information Today, Inc., 2 Auflage.

[Schaffert et al., 2007] Schaffert, S., Bry, F., Baumeister, J., and Kiesel, M. (2007). Semantic wiki. *Informatik-Spektrum*, 30(6):434–439.

[Schaffert et al., 2008] Schaffert, S., Bry, F., Baumeister, J., and Kiesel, M. (2008). Semantische wikis. In Blumauer, A. and Pellegrini, T., Herausgeber, *Social Semantic Web. Web 2.0 - Was nun?*, Seiten 23–41. Springer, Berlin Heidelberg.

[Schmidt, 2008] Schmidt, J. (2008). Geschlechterunterschiede in der deutschsprachigen blogosphäre. In und Steffen Blaschke, P. A., Herausgeber, *Web 2.0 - Eine empirische Bestandsaufnahme*, Seiten 75–86. Springer, Wiesbaden: Vieweg.

[Schuegraf and Meier, 2005] Schuegraf, M. and Meier, S. (2005). Chat- und forenanalyse. URL: `http://www.tuchemnitz.de/phil/medkom/mk/meier/schuegraf.meier_chat.forenanalyse.pdf` [Aufruf 08.10.2009].

[Schutta and Asleson, 2006] Schutta, N. T. and Asleson, R. (2006). *Pro Ajax and Java*. Apress.

[Shi et al., 2009] Shi, X., Zhu, J., Cai, R., and Zhang, L. (2009). User grouping behavior in online forums. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, Seiten 777–786, New York, NY, USA. ACM.

[Simmons, 2007] Simmons, A. (2007). Rich internet applications 101: A primer for marketing agencies and multimedia developers. White Paper. URL: `http://www.inm.com/resource-center/en/white-paper/INM-RIA-Primer.pdf`.

[Singer, 2009] Singer, A. (2009). Don't let social media comments ruin discussions on your blog. Website. URL: `http://thefuturebuzz.com/2009/08/30/social-media-comments/` [Aufruf 24.09.2009].

[SixApart, 2004] SixApart (2004). Trackback technical specification. Website. URL: `http://www.sixapart.com/pronet/docs/trackback_spec` [Aufruf 14.09.2009].

[Stern, 2006] Stern, A. (2006). Future of web apps - kevin rose. Website. URL: `http://www.centernetworks.com/future-of-web-apps-kevin-rose` [Aufruf 05.08.2009].

[Stocker and Tochtermann, 2009] Stocker, A. and Tochtermann, K. (2009). Anwendungen und technologien des web 2.0: Ein Überblick. In Blumauer, A. and Pellegrini, T., Herausgeber, *Social Semantic Web*, X.media.press, Seiten 63–82. Springer.

[Stockmann, 2004] Stockmann, G. (2004). Kommunikation in internet-foren. theorien, analysen und befunde zu forenkommunikation am beispiel des universitären elearning. In *8. Workshop der DGPuK-Fachgruppe CvK 2004: Extended Abstracts*.

[Sun Microsystems, 2010] Sun Microsystems, I. (2010). Javafx. Website. URL: `http://www.sun.com/software/javafx/` [Aufruf 16.01.2010].

[Surowiecki, 2005] Surowiecki, J. (2005). *Die Weisheit der Vielen nutzen*. Bertelsmann, München, 1. Auflage.

[Sussman, 2009] Sussman, M. (2009). State of the blogosphere 2009. Website. URL: `http://technorati.com/blogging/feature/state-of-the-blogosphere-2009/` [Aufruf 15.12.2009].

[Tele-Atlas, 2008] Tele-Atlas (2008). Google signs five year map agreement with tele atlas. Website. URL: `http://www.teleatlas.com/WhyTeleAtlas/Pressroom/PressReleases/TA_CT018846` [Aufruf 22.07.2009].

[Teorey et al., 2005] Teorey, T. J., Lightstone, S., and Nadeau, T. (2005). *Database modeling and design: logical design*. Academic Press.

[Terdiman, 2005] Terdiman, D. (2005). Study: Wikipedia as accurate as britannica. Website. URL: `http://news.cnet.com/Study-Wikipedia-as-accurate-as-Britannica/2100-1038_3-5997332.html` [Aufruf 17.07.2009].

[Toffler, 1980] Toffler, A. (1980). *The third wave*. Bantam.

[Trevino, 2005] Trevino, E. M. (2005). Blogger motivations: Power, pull, and positive feedback. *Internet Research 6.0*.

[Vickery and Wunsch-Vincent, 2007] Vickery, G. and Wunsch-Vincent, S. (2007). *Participate Web and User-Created Content: WEB 2.0, WIKIS and Social Networking*. Organization for Economic Co-operation and Development (OECD).

[Walls and Breidenbach, 2008] Walls, C. and Breidenbach, R. (2008). *Spring im Einsatz*. Hanser Fachbuchverlag.

[Walter, 2008] Walter, H.-D. (2008). Rich internet applications – eine perfekte kombination benutzerfreundlicher schnittstellen mit webtechnologie. *Informatik-Spektrum*, 31(4):333–343.

[West, 2008] West, M. L. (2008). *Using Wikis for Online Collaboration: The Power of the Read-Write Web*. John Wiley and Sons.

[WikiMatrix, 2009a] WikiMatrix (2009a). Wikimatrix - compare them all. Website. URL: `http://www.wikimatrix.org/` [Aufruf 24.09.2009].

[WikiMatrix, 2009b] WikiMatrix (2009b). Wikimatrix - search: List of wikis that support page redirection. Website. URL: `http://www.wikimatrix.org/search.php?sid=2197` [Aufruf 30.10.2009].

[WordPress, 2009] WordPress (2009). Wordpress feeds. Website. URL: `http://codex.wordpress.org/WordPress_Feeds` [Aufruf 18.09.2009].

[Zeitner et al., 2008] Zeitner, A., Linner, B., Maier, M., and Göckeler, T. (2008). *Spring 2.5: Eine pragmatische Einführung*. Addision Wesley.