UNIVERSITÄT SIEGEN
Fachbereich 5
Wirtschaftsinformatik und neue Medien
Prof. Dr. Volker Wulf

Diplomarbeit

zur Erlangung des akademischen Grades
„Diplom-Wirtschaftsinformatiker“

Entwicklung eines Suchsystems für Webservices

Ergebnispersonalisierung auf Basis community-generierter Inhalte

Autor: Thorsten Theelen
Matrikelnummer: 617325

Betreuer: Dipl.-Wirt.-Inform. Christian Dörner

Erstprüfer: Jun. Prof. Dr. Volkmar Pipek

Zweitprüfer: Prof. Dr. Volker Wulf

Einreichung: 13.02.2009

Inhaltsverzeichnis

Inhaltsverzeichnis ..... **I**

1 Einleitung **1**

2 Grundlagen **3**

2.1 Serviceorientierte Architekturen 3
2.1.1 Webservices 5
2.1.2 Schwächen serviceorientierter Architekturen 7

2.2 Ansätze zur Verbesserung des Service Discovery 9
2.2.1 Semantic Web Services 9
2.2.2 Kontextsensitives Service Discovery 12
2.2.3 Beurteilung 13

2.3 Community-generierte Inhalte und Personalisierung 14

2.4 Informationsgewinnung aus Texten 17
2.4.1 Vorbereitung von Texten 17
2.4.2 Gewichtung von Schlüsselwörtern 18

2.5 Ähnlichkeitsbestimmung 20
2.5.1 Qualität von Algorithmen zur Ähnlichkeitsberechnung 20
2.5.2 Ähnlichkeitsgraphen 22

2.6 Personalisierung 26
2.6.1 Recommender Systeme 26
2.6.2 Relevanz-Feedback 28
2.6.3 Beurteilung 29

2.7 Zusammenfassung 29

3 Systementwurf **31**

3.1 Anforderungen 31
3.1.1 Benutzeroberfläche 31
3.1.2 Berechnung der Suchergebnisse 32
3.1.3 Relevanz-Feedback und Personalisierung 33
3.1.4 Zusammenfassung 34

3.2 Entwurf 35
3.2.1 Datenbasis 36
3.2.2 Such- und Ranking-Algorithmen 40
3.2.3 Relevanz-Feedback und Personalisiertes Suchranking 52

3.3 Zusammenfassung und Beurteilung 58

4 Implementierung **60**

4.1 Die SiSeOr-Orchestrierungsumgebung 60
4.1.1 EUSOP 60

4.1.2 SiSeOr....63
4.1.3 Eclipse Rich Client Platform....64
4.1.4 Anbindung des Suchsystems....65
4.2 Datenhaltung....66
4.2.1 Graphen....68
4.3 Benutzerprofile und Relevanz-Feedback....73
4.4 Komponenten....74
4.4.1 Textmining....74
4.4.2 Suchmodule....75
4.4.3 Strategien und Konfiguration....76
4.5 Benutzeroberfläche....77

5 Evaluation....80

5.1 Ziele und Methodik....80
5.2 Setting und Ablauf....81
5.3 Qualitative Auswertung der Fragebögen....83
5.4 Beschränkungen....86

6 Zusammenfassung und Ausblick....88

Literaturverzeichnis....IV

Abbildungsverzeichnis....VIII

Tabellenverzeichnis....VIII

Anhang: Fragebögen Benutzungstest....IX

Erklärung....XXXIV

1 Einleitung

Von Unternehmen wird heutzutage erwartet, dass sie schnell auf neue Situationen reagieren und ihre Geschäftsprozesse flexibel an die neuen Gegebenheiten anpassen können. Diese Änderungen müssen sich auch in den verwendeten Informationssystemen widerspiegeln, was jedoch mit vielen der heute im Einsatz befindlichen Systemen nur schwer zu erreichen ist (vgl. Heutschi [25]). Anwendungen werden oft seit vielen Jahren verwendet. Bei ihrer Entwicklung waren heutige Anforderungen nicht absehbar und es fehlen die Möglichkeiten sie auf einfache Weise um neue Funktionen zu erweitern. Um trotzdem Anpassungen an derartigen Systemen durchführen zu können, ist häufig ein großer Programmieraufwand notwendig, was die Ausführung durch Softwareentwickler notwendig macht. Diese Entwickler arbeiten in der Regel nicht selbst als Anwender mit der von ihnen entwickelten Software, entsprechend haben sie nicht dasselbe Verständnis für die Aufgabe des Systems wie die Benutzer. Diesen fällt es jedoch häufig schwer, präzise ihre Wünsche zu formulieren, so dass Veränderungen an bestehender Software nicht immer das erfüllen, was sich die Anwender von ihnen erhofft haben. Derartige Kommunikationsprobleme zwischen Anwendern und Entwicklern und daraus möglicherweise resultierende nachträgliche Ausbesserungen können zu einem langen und aufwändigen Entwicklungsprozess führen und somit verhindern, dass Softwaresysteme schnell und flexibel an neue Anforderungen angepasst werden können.

Eine mögliche Lösung für diese Probleme stellen serviceorientierte Architekturen dar, die einzelne Programmfunktionen als so genannte Dienste anbieten, die dann flexibel zu Gesamtanwendungen zusammengefügt werden können. Diese Dienstorchestrierung erfordert keine Programmierkenntnisse, die verwendeten Technologien und Werkzeuge sind jedoch zu kompliziert, um von Benutzern ohne gute Informatikkenntnisse verwendet zu werden. Das Forschungsgebiet des End-User Developments beschäftigt sich mit der Frage, wie es Endbenutzer ohne Programmierkenntnisse ermöglicht werden kann, ihre Software selbst an ihre Bedürfnisse anzupassen, oder sogar eigene Softwareartefakte zu entwickeln (vgl. Lieberman et al.[34]). Mit SiSeOr (Simple Service Orchestration) wurde nach den Grundsätzen des End-User Developments eine Anwendung entwickelt, die es Endbenutzern ermöglichen soll, eigenständig Webservices, die eine mögliche Technologie zur Realisierung serviceorientierter Architekturen darstellen, zu orchestrieren, also aus einzelnen Diensten komplexere Anwendungen zu erstellen (vgl. Paczynski [45]). Die Verwendung von Webservices leidet allerdings unter einem großen Problem, das auch von SiSeOr nicht behoben werden konnte: Die zur Verfügung stehenden Möglichkeiten zur Suche nach

Diensten erschweren es dem Benutzer für seinen Zweck geeignete Dienste zu finden. Bevor die Dienste, die der Benutzer zur Erfüllung seines Ziels benötigt nicht gefunden werden, kann allerdings keine Orchestrierung durchgeführt werden, egal wie benutzerfreundlich die dazu verwendete Software ist. Die Schwierigkeiten beim Auffinden von Diensten behindern allgemein die Verwendung von serviceorientierten Architekturen, so dass die Idee entstand, ein neues, endbenutzergerechtes Webservice Suchsystem zu entwickeln.

Durch die zunehmende Beliebtheit des „Web 2.0“[42], sind von Benutzergemeinschaften erstellte Inhalte weit verbreitet. Von Benutzern mit Inhalten gefüllte Wiki-Systeme ersetzen zum Teil traditionelle Informationsquellen und von anderen Benutzern vergebene Schlagworte (oder „Tags“) helfen bei der Suche nach geeigneten Inhalten. Personalisierte Empfehlungen auf Basis von Benutzerprofilen, die Informationen über den Benutzer enthalten, werden z.B. in Onlineshops wie Amazon.de[5] verwendet, um dem Nutzer die Informationen anzubieten, die seinen persönlichen Interessen entsprechen. Diese Ansätze könnten auch als Teil eines Webservice-Suchsystems zur Unterstützung von Endbenutzern bei der Auswahl geeigneter Dienste beitragen, wozu sie jedoch an den konkreten Anwendungsfall der Webservice-Suche angepasst werden müssen. Die Frage, die in dieser Diplomarbeit beantwortet werden soll lautet daher:

Wie können community-generierte Inhalte und Benutzerprofile in einen Suchalgorithmus für Webservices integriert werden, um die Ergebnisqualität zu verbessern?

Um diese Frage zu klären, werden zuerst in Kapitel 2 die Grundlagen serviceorientierter Architekturen und der konkret verwendeten Webservices erklärt, um die Probleme aktueller Verfahren zur Suche nach Webservices herauszuarbeiten. Daraufhin werden vorhandene Ansätze zur Verbesserung der Dienstsuche vorgestellt und es wird beschrieben was genau mit community-generierten Inhalten und Personalisierung gemeint ist und wie heutige Systeme, in denen sie angewendet werden funktionieren. In Kapitel 3 wird ein auf Basis dieser Erkenntnisse entwickeltes Webservice Suchsystem vorgestellt. Kapitel 4 beschäftigt sich mit den Implementierungsdetails des Systems und Kapitel 5 beschreibt einen Benutzungstest, der zur Evaluation des Suchsystems durchgeführt wurde.

2 Grundlagen

In diesem Kapitel werden die Grundlagen serviceorientierter Architekturen erklärt. Ziel des Kapitels ist es, Schwierigkeiten bei der Suche nach Diensten unter Verwendung aktueller Technologien herauszuarbeiten und anhand vorhandener Ansätze die Basis zur Klärung der Fragestellung dieser Arbeit zu bilden.

2.1 Serviceorientierte Architekturen

Betriebliche Informationssysteme müssen flexibel an neue Situationen angepasst werden können, um auf die sich ständig ändernden Anforderungen an Unternehmen reagieren zu können. Diese Anforderung wird jedoch durch vorhandene, häufig seit vielen Jahren im Einsatz befindliche Systeme nicht immer erfüllt. Die verwendeten Technologien sind oft veraltet und es existieren keine Schnittstellen zur Anbindung neuer Applikationen, wodurch ein Austausch von einzelnen Teilanwendungen oder die Integration neuer Applikationen aufwändig und teuer ist. Des Weiteren können derartige Anpassungen zu immer komplizierteren Systemen mit redundanten Daten und Funktionen führen, deren Wartung und Erweiterung mit der Zeit immer schwieriger wird. Nach Heutschi [25] können diese Probleme durch die Heterogenität vorhandener Systeme in den folgenden Bereichen verursacht werden:

- *Plattformabhängigkeit:* Hardware und Betriebssysteme, auf denen die zu verknüpfenden Anwendungen basieren, können sich unterscheiden.
- *Unterschiedliche Datenmodelle und Datenformate:* Gleichbedeutende Konzepte können von verschiedenen Anwendungen auf unterschiedliche Weise dargestellt, gespeichert und ausgetauscht werden.
- *Unterschiedliche Funktionsmodelle:* Funktionen einer Anwendung erfüllen eine Aufgabe nicht auf genau die Weise die eine andere erwartet, liefern also ggf. keine verwendbaren Ergebnisse.
- *Prozessabhängigkeit:* Eine Software realisiert häufig einen vollständigen Geschäftsprozess. Die Änderung oder der Austausch einer einzelnen Komponente ist nicht vorgesehen, somit sind die Auswirkungen auf den Gesamtprozess nicht vorhersehbar.

Serviceorientierte Architekturen (SOA) bieten eine mögliche Lösung für dieses Problem. Sie setzen sich aus lose gekoppelten, wiederverwendbaren Diensten zusammen und ermöglichen so eine Komponenten-basierte Entwicklung von Softwaresystemen (vgl. Brahe und Schmidt [12]). Das Ziel besteht dahin, die Schwächen bisheriger Systemarchitekturen zu beheben, indem einzelne Teilaufgaben gekapselt und als Dienste angeboten werden. Diese Dienste verfügen über standardisierte, klar festgelegte Schnittstellen, die es ermöglichen, eine Gesamtfunktionalität als Verknüpfung verschiedener Dienste zu realisieren. Solange die Schnittstelle unverändert bleibt lassen sich einzelne Komponenten austauschen.

Dienste können einer Teilaufgabe innerhalb eines Geschäftsprozesses entsprechen. So kann aus dem Modell eines Geschäftsprozesses eine Abfolge von Dienstaufrufen abgeleitet werden, die zusammen ein flexibel an neue Situationen anpassbares Informationssystem ergeben.

Es gibt verschiedene Definitionen für serviceorientierte Architekturen. Die hier verwendete, stammt von Heutschi [25].

> *„Eine SOA ist eine mehrschichtige, verteilte Informationssystem (IS)-Architektur, die Teile von Applikationen für eine vereinfachte Prozessintegration als geschäftsorientierte Services kapselt und dabei bestimmte Designprinzipien berücksichtigt.*
>
> *Ein Service stellt ein abstraktes Software-Element bzw. eine Schnittstelle dar, die anderen Applikationen über ein Netzwerk einen standardisierten Zugriff auf Anwendungsfunktionen anbietet.“*

Benutzer eines Anwendungssystems interessieren sich in der Regel nicht für die Implementierungsdetails der Software, sondern für die angebotene Funktionalität in Zusammenhang mit den für sie relevanten Geschäftsprozessen (vgl. Hofmann et al.[26]). In Diensten werden Funktionen mit einer von der Implementierung unabhängigen Schnittstelle versehen. Um einen Dienst zu benutzen, muss der Benutzer verstehen, welche Funktionen angeboten werden, allerdings nicht wie diese genau im Programm umgesetzt sind. Dies entspricht der Denkweise der Anwender und erleichtert ihnen somit die Nutzung der Dienste.

In serviceorientierten Architekturen unterscheidet man zwischen drei wesentlichen Rollen, die in Abbildung 2.1 zu sehen sind. Der *Serviceanbieter* stellt den Dienst und die dazugehörige Dienstbeschreibung zur Verfügung. Diese Beschreibung definiert welche Funktionen ein Dienst anbietet und wie mit ihm kommuniziert werden kann. Zur Veröffentlichung dieser Servicebeschreibungen dient das *Serviceverzeichnis*, welches dem *Servicenutzer* die Möglichkeit bietet nach Diensten zu suchen und ihm die benötigten Informationen zur Nutzung des Dienstes liefert.

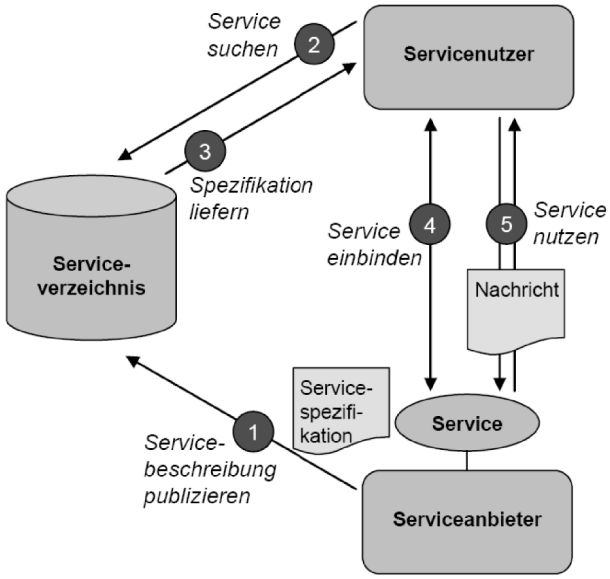

Abbildung 2.1: SOA-Interaktionsdreieck ([25])

Die wichtigsten Design-Grundsätze von serviceorientierten Architekturen sind nach Heutschi [25]:

- Schnittstellenorientierung: Die tatsächliche Implementierung rückt in den Hintergrund, der Nutzer arbeitet mit abstrakten Diensten.
- Interoperabilität: Alle Teile einer SOA halten sich bezüglich Schnittstellenbeschreibung, Kommunikation etc. an festgelegte Standards
- Autonomie und Modularität: Das Gesamtsystem wird in voneinander unabhängige Teilkomponenten aufgeteilt
- Bedarfsorientierung: Dienste werden so eingeteilt, dass sie eine im Kontext der zu realisierenden Prozesse sinnvolle Funktionseinheit darstellen

2.1.1 Webservices

Bisher wurde der allgemeine Aufbau von serviceorientierten Architekturen dargestellt. Der folgende Abschnitt beschäftigt sich mit Webservices, die den verbreitetsten Ansatz zu deren Realisierung darstellen. Sie basieren auf einer Reihe von auf XML (Extensible Markup Language [18]) basierenden Standards, die dazu dienen, Dienste zu beschreiben und deren Kommunikation über ein Netzwerk zu ermöglichen. Die Interaktion zwischen Webservices gleicht dabei entfernten Prozeduraufrufen. Ein Dienst bietet eine Prozedur an, die von einem

anderen Dienst über ein Netzwerk aufgerufen werden kann und sendet, wenn nötig, eine Antwort zurück. Im Folgenden werden die wesentlichen Bestandteile der Webservice-Architektur vorgestellt.

Zur Beschreibung von Diensten dient die „Web Services Description Language“ (WSDL [67]). WSDL ist eine XML-basierte Sprache, die Möglichkeiten bietet, alle zur Sicherstellung der Interoperabilität von Diensten benötigten Informationen darzustellen (vgl. Wen und He [63]). Sämtliche Schnittstellen, die ein Dienst anbietet, werden in WSDL-Dateien definiert, d.h. alle Operationen, auszutauschende Nachrichten und deren Attribute. Des Weiteren werden Nachrichtenformate festgelegt und es besteht die Möglichkeit, einzelne Elemente der Dienstbeschreibung über ein „<documentation>“-Element mit natürlichsprachlichen Dokumentationen zu versehen. Diese werden nicht automatisch ausgewertet, sondern können für den Dienst-Nutzer bestimmte Informationen über die Funktionsweise des Dienstes enthalten.

Als Serviceverzeichnis wird bei Webservices der UDDI-Standard (Universal Description Discovery and Integration [60]) genutzt. UDDI bietet ein Verzeichnis, in dem Serviceanbieter ihre Dienste veröffentlichen können. UDDI-Einträge enthalten grundlegende Informationen über den Dienst und den Anbieter. Im Falle des Anbieters handelt es sich unter anderem um Namen, Kontaktdaten, einen Beschreibungstext und Referenzen auf die Dienste des Anbieters sowie beispielsweise Webseiten und andere Quellen mit Bezug auf den Anbieter. Diese Einträge werden in UDDI „businessEntity“ genannt. Diensteinträge („businessService“) enthalten den Namen des Dienstes, eine Beschreibung sowie Verweise auf die technische Beschreibung des Dienstes in der „tModel“-Datenstruktur, die unter anderem auf die WSDL-Beschreibung des Dienstes verweisen kann. Die verwendeten Felder sind wie auch das „tModel“ zum großen Teil komplexe Datenstrukturen, deren Inhalt nicht klar definiert ist, und sich in verschiedenen Verzeichnissen, oder auch unterschiedlichen Einträgen desselben Verzeichnisses unterscheiden kann. Sämtliche Einträge im UDDI-Verzeichnis verfügen über einen „Universally Unique Identifier“ (UUID) zur eindeutigen Identifikation, der beim Einfügen in das Verzeichnis generiert wird. Zum Auffinden von Diensten bietet UDDI eine Schlüsselwortsuche, die auf beliebige Textelemente innerhalb der in UDDI vorhandenen Datenstrukturen zugreifen kann.

Die Entwicklung von UDDI wurde von Unternehmen wie Microsoft, IBM, SAP u.a. vorangetrieben, die gemeinsam die UBR (UDDI Business Registry) betrieben, ein globales, öffentliches UDDI-Verzeichnis, in dem jeder Anbieter seine Dienste veröffentlichen konnte. Dieses Verzeichnis wurde Anfang 2006 mit der Begründung geschlossen, dass es keine feste Einrichtung, sondern eine Testplattform für die Möglichkeiten von UDDI war und diese Aufgabe erfüllt hat (s. UBR Shutdown FAQ[37]). Dieser Schritt wurde von vielen Seiten als

das Ende von UDDI gedeutet und als schwerer Schlag für serviceorientierte Architekturen allgemein betrachtet (z.B. X. Shi[55]). Die hauptsächliche Nutzung von UDDI liegt seitdem bei voneinander isolierten Service-Verzeichnissen, beispielsweise innerhalb eines Unternehmens.

Die Kommunikation zwischen Diensten geschieht über SOAP [56], ein XML-basiertes Dateiformat, das den Inhalt von Nachrichten zwischen Diensten enthält. Die eigentliche Übertragung dieser Nachrichten wird dabei nicht vorgegeben, sie geschieht i.d.R. über vorhandene Übertragungsprotokolle wie z.B. HTTP.

Ein Ziel von serviceorientierten Architekturen ist die Orchestrierung von Diensten zur Nachbildung von Geschäftsprozessen in Unternehmen. Dies sollte, wenn möglich, innerhalb der Unternehmen geschehen können und keinen großen Programmieraufwand erfordern. Zu diesem Zweck gibt es BPEL (Business Process Execution Language [62]), eine ebenfalls auf XML basierende Sprache, mit der eine Ablaufreihenfolge von Dienstaufrufen festgelegt werden kann. Eine so zusammengesetzte Dienstkomposition erscheint nach außen hin wiederum als Dienst. BPEL regelt die Kommunikation zwischen verschiedenen Diensten, nicht aber die Einbeziehung von menschlichen Nutzern in den Prozessablauf. Diesen Zweck soll die Erweiterung BPEL4People [66] erfüllen . Da die manuelle Erstellung von BPEL-Dateien aufwändig und fehleranfällig ist, existieren diverse grafische Editoren, die den Benutzern die Orchestrierung erleichtern sollen (z.B. Active VOS[2], Eclipse BPEL-Project[16]).

2.1.2 Schwächen serviceorientierter Architekturen

Die Vorteile von serviceorientierten Architekturen können genutzt werden, um Endbenutzer in den Entwicklungsprozess einzubeziehen. Der spätere Nutzer hat eine andere Sicht auf das zu lösende Problem als der Entwickler, seine Einbeziehung kann dabei helfen, mögliche Probleme frühzeitig zu erkennen und zu vermeiden, sowie Anpassungsvorgänge zu beschleunigen. Aktuell verwendeten Technologien, also insbesondere XML-basierte, für die automatisierte Kommunikation zwischen Diensten vorgesehene Sprachen, aber auch grafische Editoren (z.B. für BPEL) sind zu komplex um von einem durchschnittlichen Computernutzer erlernt und genutzt zu werden (vgl. Brahe u. Schmidt[12], Liu et al.[35]).

Das Auffinden von geeigneten Diensten („Service Discovery“) stellt bei den derzeit genutzten Technologien eine weitere Schwierigkeit dar. Ständig werden neue Web Services entwickelt und veröffentlicht, häufig mit vergleichbarer Funktionalität, dadurch ist es wichtig, die Qualität eines Dienstes beurteilen zu können (vgl. Abramowicz et al.[1]). So kann es z.B. sein, dass ein Dienst, der nicht in der Lage ist seine Aufgabe innerhalb einer vorgegebenen Zeit zu

erfüllen, nutzlos ist, was durch eine Integration derartiger Informationen in den Suchvorgang berücksichtigt werden könnte. Auch das Einbeziehen von Kontextinformationen über den Dienst und den Nutzer können zur Verbesserung des Service Discovery beitragen (vgl. Kuck u. Reichartz[33]). Bietet ein Dienst beispielsweise Zugriff auf einen Drucker, so ist dessen Nutzung nur sinnvoll, falls dieser vom Nutzer aus erreichbar ist, sich also z.B. im selben Gebäude befindet. Die Suchfunktionen von UDDI reichen hierfür nicht aus, da sie nur eine einfache Schlüsselwortsuche nach Dienstnamen und anderen im UDDI gespeicherten Informationen ermöglichen. Angaben über die Qualität des Dienstes sind nicht enthalten, dem Benutzer wird somit keine Unterstützung bei der Wahl zwischen austauschbaren Diensten geboten (vgl. Bianchini et al.[11]). Zwar lassen sich über die komplexen Datenstruktur in UDDI weiterführende Informationen an UDDI-Einträge anbinden, die Suchfunktion greift allerdings nur auf die Textinformationen innerhalb dieser Einträge zu und nicht auf die weiterführenden Informationen auf die verwiesen wird. Wenn z.B. eine Referenz auf eine WSDL-Datei vorliegt, so kann zwar die URL zu dieser Datei in die Schlüsselwortsuche einbezogen werden, der Inhalt auf den verwiesen wird, steht jedoch nicht zur Verfügung.

Um das Verzeichnis auf einem aktuellem Stand zu halten, ist es nötig, dass Dienstanbieter alle Änderungen, also z.B. das Wegfallen eines Dienstes sofort verzeichnen, was kaum sicherzustellen ist. Des Weiteren überprüft UDDI nicht, ob eingetragene Informationen tatsächlich zu einem gültigen Dienst gehören. Es ist theoretisch möglich, beliebige Informationen ohne Bezug zu Web Services in UDDI zu speichern. Demnach lässt sich nicht garantieren, dass alle Einträge im UDDI auch einen tatsächlich existierenden Dienst beschreiben bzw. dass die enthaltenen Informationen noch aktuell sind (vgl. Bachlechner et al.[8]).

WSDL bietet zwar die Möglichkeit, Dienste zu dokumentieren, diese wird jedoch kaum genutzt und die Form der Dokumentation ist nicht einheitlich. Hier taucht außerdem ein weiteres Problem auf. Die verwendete Sprache kann sich aufgrund der unterschiedlichen Perspektive von Benutzer und Entwickler unterscheiden. Benutzer suchen nach einem Weg zur Erfüllung ihres Ziels, sie formulieren eine Suchanfrage beispielsweise im Kontext eines bestimmten Geschäftsprozesses. Der Anbieter beschreibt allgemein die Funktionalität des Dienstes ohne einzelne Anwendungsfälle zu berücksichtigen. Diese beiden Beschreibungen können sich syntaktisch unterscheiden, auch wenn die Semantik dieselbe ist, eine Schlüsselwortsuche kann allerdings nur erfolgreich sein, wenn die in der Dokumentation verwendeten Begriffe mit den späteren Suchbegriffen übereinstimmen (vgl. Aguilera et al. [3]).

Die vorgestellten Schwächen serviceorientierter Architekturen, insbesondere im Bereich des

Service Discovery erschweren deren Nutzung. In den folgenden Abschnitten werden daher verschiedene Ansätze vorgestellt, die sich mit der Lösung dieser Probleme beschäftigen.

2.2 Ansätze zur Verbesserung des Service Discovery

Wie im vorherigen Abschnitt festgestellt wurde, bietet der UDDI-Standard nicht die nötigen Funktionen um eine effektive Suche nach Webservices zu ermöglichen. Insbesondere wenn es darum geht, Endbenutzer bei der Suche nach Diensten zu unterstützen sind zur Zeit Internet-Suchmaschinen und auf Webservices spezialisierte Web-Portale die beste Lösung (vgl. Bachlechner et al. [8]). So bieten auf Webservices spezialisierte Webseiten wie z.B. Seekda.com[53], aber auch Domänen-unabhängige Suchmaschinen wie Google[22] Möglichkeiten zur Suche nach Webservices.

2.2.1 Semantic Web Services

Das heutige World Wide Web dient dazu, Informationen auf eine für Menschen lesbare Art und Weise darzustellen. Dies gilt auch für Web-basierte Suchmaschinen, was die Anbindung an Web-unabhängige Orchestrierungsumgebungen erschwert, da notwendige Schnittstellen häufig nicht vorhanden sind. Die automatische Auswertung von Webseiten beschränkt sich meistens auf zur Darstellung der Seite benötigte Daten. Die Semantik, der auf der Webseite angebotenen Informationen kann nicht verarbeitet werden. Das Semantic Web soll das heutige Web um eine zusätzliche Metadaten-Ebene erweitern, die es Computerprogrammen erlaubt die Semantik zu erfassen und zur selbstständigen Ausführung komplexer Aufgaben für Benutzer zu verwenden, was eine automatisierte Nutzung von Webservices nötig machen kann. Die Idee zum Semantic Web stammt von Tim Berners-Lee[1] [9].

Zur Darstellung von Informationen im Semantic Web werden auf XML basierende, und somit Computer-lesbare Sprachen verwendet. Um aus Daten Rückschlüsse über deren Bedeutung ziehen zu können, reicht Lesbarkeit alleine noch nicht aus. Zur Beschreibung der dazu benötigten Semantik kann RDF (Resource Description Framework[51]) verwendet werden. Um ein Objekt in RDF zu beschreiben, wird ein Tripel von URIs (Uniform Resource Identifier), also beispielsweise Verweise auf andere Web-Ressourcen verwendet. Diese drei URIs übernehmen dabei die Funktion von Subjekt, Prädikat und Objekt in einem Satz. Es wird davon ausgegangen, dass zu beschreibende Objekte Eigenschaften mit Werten haben. Der erste URI verweist auf das Subjekt der Beschreibung, der zweite auf eine Definition der Eigenschaft und der dritte auf das Objekt, das den Wert der Eigenschaft darstellt. Da ein URI

1 Begründer des World Wide Web (s. http://www.w3.org/People/Berners-Lee/Longer.html)

immer auf die Definition eines bestimmten Konzeptes verweist ist auch bei mehrdeutigen Begriffen die beabsichtigte Semantik erkennbar.

Da verschiedene Informationsquellen unterschiedliche Begriffe verwenden können, verwendet das Semantic Web Ontologien, um Beziehungen zwischen verwandten Konzepten darzustellen. Der Begriff „Ontologie“ stammt ursprünglich aus der Philosophie, in der Informatik bezeichnet man damit ein Dokument oder eine Datei, die das Wissen über eine Domäne modelliert. Sie enthält dabei üblicherweise Klassen von Objekten, Attribute und Beziehungen zwischen diesen Objekten und Regeln die diese Beziehungen beschreiben (vgl. Gruber[24], Berners-Lee[9]). Man kann Ontologien als gerichteten Graphen verstehen, in dem die Kanten für semantische Beziehungen zwischen Konzepten stehen. Durch die Beziehungs-Typen und die gerichteten Kanten besteht eine hierarchischen Beziehung zwischen den Elementen. Eine mögliche Anwendung für Ontologien ist die Erweiterung von Suchmaschinen, so dass nicht nur nach syntaktischen Übereinstimmungen mit Schlüsselwörtern, sondern auch nach semantisch verwandten Konzepten gesucht wird. Zum Aufbau von Ontologien im Semantic Web wird die „Web Ontology Language“ (OWL[43]) benutzt. Ein Beispiel für eine öffentlich verfügbare Ontologie ist WordNet[65], ein englischsprachiges, semantisches Wörterbuch, in dem Beziehungen zwischen Begriffen dargestellt werden, um beispielsweise Synonyme eines Wortes abzufragen. EuroWordNet[17] erweitert dieses Prinzip um mehrere europäische Sprachen.

Das Semantic Web kann als Basis für Agentensysteme dienen, also Programme die stellvertretend für einen menschlichen Benutzer, selbstständig Aufgaben ausführen und dazu auf Web-Ressourcen zugreifen. Die Zahl an verfügbaren Webservices nimmt immer weiter zu, allerdings fehlt ihnen eine semantische Beschreibung, die ein Auffinden der Dienste und eine Nutzung durch Agentensysteme erlauben würde. Aus diesem Grund gibt es Bestrebungen, die Idee des Semantic Webs auf Webservices zu übertragen, um so eine automatisierte Suche, Auswahl, Ausführung und Komposition von Webservices zu ermöglichen (vgl. Burstein et al.[13]). Derartige semantische Beschreibungen könnten auch genutzt werden, um Service-Discovery Funktionen für menschliche Benutzer zu verbessern(vgl. Aiken[4]).

Es gibt verschiedene Ansätze, wie die Verbindung zwischen den bestehenden Webservice Technologien und dem Semantic Web aussehen kann. Das Grundprinzip ist aber immer, die vorhandenen Dienstbeschreibungen durch zusätzliche, semantische Beschreibungen zu ergänzen, die vom Dienstanbieter bei der Veröffentlichung festgelegt werden. Diese verweisen auf Ontologien, die Informationen über im Dienst verwendeten Konzepte bieten. Dabei stehen die funktionalen Aspekte wie Operationsnamen und Attribute, sowie Qualitätsmerkmale im Vordergrund, so dass Dienste gefunden werden können die genau

den Anforderungen des Suchenden entsprechen.

Ein Beispiel für ein solches Verfahren stammt von Paolucci et al.[46] [47]. Hier wird ein auf OWL-S[2](Web Ontology Language for Webservices[44]) basierendes Service-Profil verwendet um Dienste semantisch zu beschreiben. OWL-S-Profile dienen dazu, verschiedene Dienst-Attribute an Ontologien anzubinden, die alle für die Domäne des Webservices relevanten Konzepte definieren. Diese Profile werden von den Dienstanbietern erstellt und einer Matching-Engine übergeben, die später für die semantische Suchfunktion zuständig ist. Zusätzlich wird aus dem Profil ein UDDI-Eintrag generiert. Einige Felder aus dem Profil wie z.B. Dienst- und Anbieternamen lassen sich direkt in UDDI abbilden, andere Informationen, die UDDI nicht direkt enthält werde als UDDI-tModels gespeichert. Der Vorteil dieses Vorgehens ist, dass die Möglichkeit besteht, an der semantischen Suche vorbei eine Standard-UDDI-Suche durchzuführen, ohne ein zusätzliches Dienstverzeichnis und somit auch eine weitere Anmeldung des Dienstes zu erfordern.

Eine Suchanfrage besteht aus der semantischen Beschreibung der gewünschten Funktionalität in Form eines OWL-S-Profils. Diese abstrakte Dienstbeschreibung wird von der Matching-Engine mit bekannten Diensten verglichen, indem anhand der Ontologie Übereinstimmungen zwischen den Attributen gesucht werden. Als Suchergebnisse werden die Dienste geliefert, die dem gesuchten Dienstprofil am ähnlichsten sind. Die Ontologie ermöglicht es aufgrund der in ihr abgebildeten semantischen Zusammenhänge auch dann Treffer zu finden wenn keine Dienstbeschreibung exakt dem gewünschten Dienst entspricht. Suchergebnisse werden in Form von den zuvor beschriebenen, eindeutigen UDDI-Dienstschlüsseln geliefert, anhand derer alle zur Nutzung eines Dienstes benötigten Informationen aus dem UDDI-Verzeichnis abgefragt werden können.

Ein ähnliches System wird von Kawamura et al.[29] vorgestellt. Service-Profile liegen hier als „Web Service Semantic Profile“ (WSSP) vor, einem von OWL-S inspirierten Format. Der Unterschied zum vorherigen Beispiel liegt darin, dass die Attribute des Profils hier nicht UDDI-Attributen entsprechen, sondern zur semantischen Beschreibung von WSDL-Dateien dienen. So gibt es zu jedem relevanten WSDL-Attribut im WSSP einen Verweis auf das dazugehörige Konzept in einer Ontologie. Zusätzlich zur semantischen Suche anhand der funktionalen Aspekte des Dienstes werden bei diesem System eine Reihe von Filter-Mechanismen zur Verbesserung der Suchergebnisse angewendet. Unter anderem wird ein Text-Filter verwendet, der auf Basis der TF-IDF-Methode (s. Kapitel 2.4.2) für Menschen lesbare Metadaten, wie z.B. Dienstbeschreibungen und Kommentare auswertet und so auch Zusammenhänge zwischen Diensten findet die sich nicht anhand vergleichbarer Operationen

2 Paolucci et al. verwenden die ursprüngliche Bezeichnung „DAML-S“ (DARPA Agent Markup Language for Web Services)

und Attribute erkennen lassen.

Ein weiterer wichtiger Aspekt der Service Discovery ist die Dienstgüte (auch: „Quality of Service“, QoS). Hierbei geht es im Zusammenhang mit Webservices darum Kriterien für die Qualität eines Dienstes aufzustellen. Dies ist beispielsweise wichtig wenn eine Entscheidung zwischen mehreren Diensten mit demselben Leistungsangebot getroffen werden muss, oder es Anforderungen gibt die auf jeden Fall erfüllt sein müssen damit ein Dienst in Frage kommt, wie z.B.die maximalen Kosten für die Nutzung des Dienstes. Derartige Kriterien unterscheiden sich abhängig von der Art und Aufgabe des Dienstes, und müssen von einer mit der jeweiligen Domäne vertrauten Community festgelegt werden (vgl. Bianchini et al. [11]). Diese Attribute können entweder statisch sein (z.B. Kosten, Standort), oder bei der Verwendung der Dienste aufgezeichnet werden (z.B. Verbindungsgeschwindigkeit, Laufzeit). Einige dieser Kriterien können eine Nutzung grundsätzlich ausschließen, während andere nur zu einem Vorzug von Alternativen führen. Das Interessante an diesem Ansatz ist, dass Webservices mit zusätzlichen Metadaten versehen werden, die in die Auswahl passender Dienste einbezogen werden.

2.2.2 Kontextsensitives Service Discovery

Neben der Dienstgüte können auch Kontext des Dienstes und des Benutzers für die Entscheidung, ob ein Dienst in Frage kommt oder nicht von Bedeutung sein. Ein Beispiel dafür wäre ein Webservice, der einen Drucker steuert. Die Verwendung dieses Dienstes macht nur Sinn, wenn sich der Nutzer im selben Gebäude befindet wie der Drucker, entsprechend sollte ein Service Discovery System diesen Dienst nur als Ergebnis liefern wenn dies zutrifft.

Kuck et al.[32][33] stellen ein System vor, das diese Anforderungen erfüllen soll. Speziell geht es um die Verwendung von Webservices auf Mobilen Geräten wie z.B. Mobiltelefonen und Laptops. So sollen den Nutzern abhängig davon, in welcher Situation sie sich gerade befinden, andere Dienste empfohlen werden. Dabei wird zwischen Benutzer- und Service-Kontext unterschieden. Der Benutzerkontext besteht aus personenbezogenen Daten (z.B. Geschlecht, Geburtsdatum), der aktuellen Situation des Nutzers (z.B. Aufenthaltsort, Datum/ Uhrzeit, Wetter) sowie Dokumenten des Nutzers, die Rückschlüsse auf seine Interessen erlauben. Diese Informationen werden als Schlüsselwörter in einem Benutzerprofil gespeichert. Das Serviceprofil besteht aus funktionalen, aus der WSDL-Beschreibung extrahierten Daten, sowie Feedback-Informationen. Diese bestehen aus Kontextinformationen von Nutzern, denen der Dienst bisher empfohlen wurde, Suchanfragen die zur Empfehlung geführt haben sowie der Akzeptanz des Dienstes, die sich aus

Relevanz-Bewertungen der Nutzer berechnet. Bei einer Suchanfrage werden die Suchbegriffe des Nutzers durch dessen aktuelle Kontext-Informationen ergänzt und diese dann mit dem Service-Profil verglichen. Über den Feedback-Mechanismus lernt das System, welchen Kontext Benutzer haben, für die ein Dienst in der Vergangenheit von Interesse war. Der Unterschied zur Betrachtung der Dienstgüte ist, dass dort in der Regel die Suchkriterien bei der Suchanfrage manuell angegeben werden, während das hier betrachtete kontext-sensitive System auf ein vorhandenes Benutzerprofil zugreift, das bei einer Suchanfrage mit den Diensten abgeglichen wird, um an Benutzerbedürfnisse angepasste Ergebnisse zu liefern.

2.2.3 Beurteilung

Die Qualität Semantic Web-basierter Verfahren hängt stark vom Aufbau der verwendeten Ontologie und der in ihr enthaltenen Informationen ab. Bernstein et al.[10] haben beim Vergleich von Suchmethoden für Ontologien erhebliche Unterschiede zwischen ihren eigenen, und den Ergebnissen einer fremden Studie festgestellt, die eine andere Ontologie nutzte. Bevor ein derartiges System funktionieren kann, muss zuerst ein Modell zur Repräsentation der benötigten Informationen innerhalb einer Ontologie erarbeitet werden. Dieses Modell ist abhängig von der betrachteten Wissensdomäne und entscheidet später über die Qualität der Daten, die von Systemen, die auf dieser Ontologie aufbauen, geliefert werden. Des Weiteren müssen die Informationen innerhalb einer Ontologie ständig aktualisiert und erweitert werden, um ein zutreffendes Bild über die jeweilige Domäne zu vermitteln.

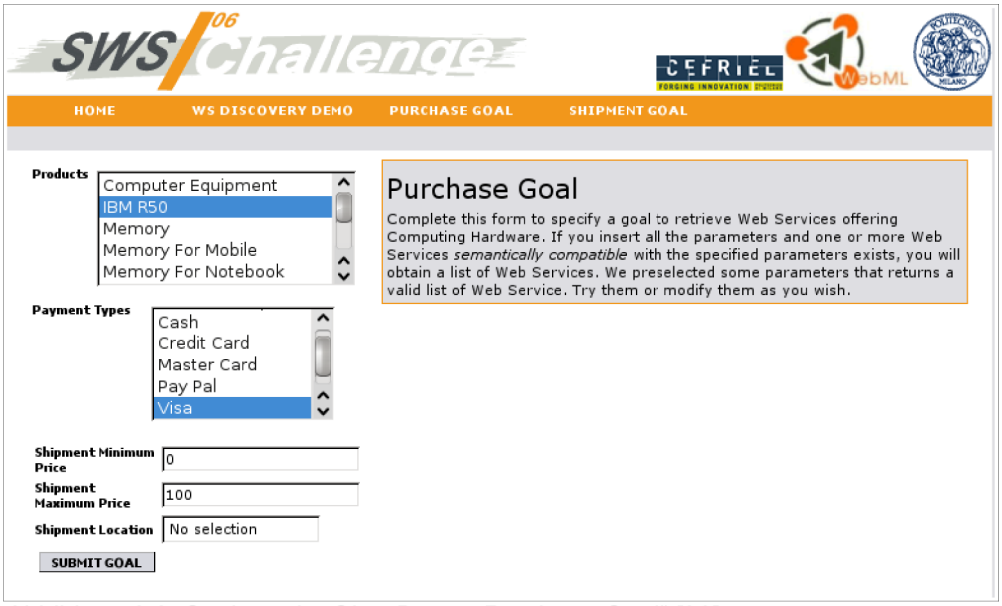

Abbildung 2.2: Suchmaske Glue-Demo „Purchase Goal“ [20]

Das Ziel von Semantic Webservices ist es, eine automatische, dynamische Service-orchestrierung durch Agentensysteme, ohne ein manuelles Eingreifen durch den Benutzer zu ermöglichen. Suchanfragen definieren präzise welche Anforderungen an einen Dienst gestellt werden, und es wird ein Dienst gesucht, der genau diese Anforderungen erfüllt. Die Verwendung von Ontologien ermöglicht es in diesem Zusammenhang lediglich, dass bei der Beschreibung nicht der exakte Wortlaut des Anbieters verwendet werden muss, sondern automatisch semantische Übereinstimmungen gefunden werden. Wie ein solches System in der Praxis aussehen kann, zeigt Glue[20], ein Java-basiertes Semantic Web Service Discovery System. Abbildung 2.2 zeigt die Suchmaske einer Demonstrations---Anwendung auf Glue-Basis, die auf der Webseite genutzt werden kann. Die gezeigte Eingabemaske gibt Suchkriterien für Dienste zur Erfüllung einer speziellen Aufgabe vor. Um nach beliebigen Diensten zu suchen, muss man sich spezieller Regel- oder Abfragesprachen bedienen. Das Erlernen einer solchen Sprache, oder die Verwendung eines Regel-Editors, wie er z.B. von Kawamura et al.[29] verwendet wird, setzt voraus, dass die Endbenutzer in der Lage sind, ihre Bedürfnisse präzise zu formulieren, was in der Realität nicht zutrifft (vgl. Shanfeng et al. [54]).

Der Kontext-sensitive Ansatz zur Service Discovery verwendet Benutzerprofile und ein Feedback-System um personalisierte Suchergebnisse zu liefern. Dieses Vorgehen lässt sich allgemein auf verschiedene Arten von Benutzerprofilen übertragen und bietet daher einen interessanten Ansatz zur Personalisierung von Suchergebnissen innerhalb von Service Discovery Systemen.

2.3 Community-generierte Inhalte und Personalisierung

Bevor ein Web Service von einem Endbenutzer verwendet werden kann, muss dieser zuerst verstehen, was der Dienst genau leistet und wie er verwendet werden kann. Wie bereits festgestellt wurde, erfüllen Dienstbeschreibungen durch den Service-Anbieter diese Anforderung nur unzureichend, da sie eine andere Perspektive repräsentieren und nicht immer in ausreichender Form vorhanden sind. Die Entwicklung und Anpassung von Software ist heute schon in vielen Fällen eine kollaborative Tätigkeit. Benutzer tauschen Erfahrungen im Umgang mit einer Software oder auch fertige Softwareartefakte untereinander aus. Ein häufiger auftretender Fall sind „Shared Use“-Szenarien, in denen einzelne Benutzer einer Software z.B. Internetforen verwenden, um Hilfe bei der Lösung ihrer individuellen Probleme zu erhalten oder ihre Erfahrungen anderen Benutzern zur Verfügung zu stellen (vgl. Pipek und Kahler[48]). Dieses Prinzip lässt sich auf Webservices übertragen. Benutzer, die bereits Erfahrungen mit einem Dienst gesammelt haben, können ihr Wissen in Form von

Beschreibungstexten, Schlagworten und anderen Metadaten für andere Benutzer zugänglich machen. Derartige Informationen machen allerdings nicht nur die Funktionsweise des Dienstes für Endbenutzer verständlicher, sie können auch als Datenbasis für eine automatische Suchfunktion dienen.

Ein Wissensaustausch innerhalb von Benutzergemeinschaften lässt sich im World Wide Web beobachten. Noch vor einigen Jahren hatten Endbenutzer außerhalb von speziellen Forensystemen und Newsgroups kaum Möglichkeiten eigene Inhalte im Web zu veröffentlichen, ohne sich in die dazu benötigten, schwer erlernbaren Technologien einzuarbeiten (vgl. Kolbitsch u. Marer[31]). Aus diesem Grund stammte lange ein Großteil der Informationen im Web von Unternehmen und anderen professionellen Anbietern. Dies hat sich in letzter Zeit geändert, immer mehr Internet-Seiten bieten ihren Besuchern die Möglichkeit, mit einfachen Werkzeugen eigene Inhalte zu veröffentlichen und an der Gestaltung von Webseiten mitzuarbeiten. Allgemein fasst man diese Art von gemeinschaftlich erstellten Web-Inhalten unter dem Begriff „Web 2.0“[42] zusammen. Web-Communities sind selbstorganisierende Strukturen, die von Kolbitsch und Maurer[KM05] mit Ameisenstaaten verglichen werden. Dort gibt es keine zentrale Aufgabenverteilung und Kontrolle, vielmehr kommunizieren die individuellen Tiere miteinander und führen die Aufgaben aus, die sie selbst für sinnvoll halten. Trotz dieses scheinbar nicht organisierten Vorgehens, scheint es nach außen so, als ob das gesamte Volk koordiniert zur Erfüllung eines gemeinsamen Ziels zusammenarbeiten würde. Wie bei den Ameisen gilt aber auch bei kollaborativen Web-Inhalten, dass eine solche Gemeinschaft eine Mindestzahl an Teilnehmern benötigt, um zu funktionieren.

Ein Beispiel für derartige Web-Communities sind Wikis. Dabei handelt es sich um Webseiten, auf denen jeder Leser selbst zum Autor werden kann, indem er eigene Beiträge verfasst oder vorhandene bearbeitet. Das bekannteste Beispiel für ein Wiki ist Wikipedia[64], eine frei verfügbare Enzyklopädie deren Einträge vollständig von den Benutzern stammen. Die Annahme ist, dass sich viele Nutzer, die Beiträge erstellen, ergänzen und gefundene Fehler korrigieren, langfristig auf vollständige und korrekte Inhalte einigen. Ein Problem von Wikis ist allerdings, dass nur ein Bruchteil der Besucher die Möglichkeit nutzt, die Seite zu bearbeiten. Viele wissen gar nicht, dass diese Möglichkeit besteht und entsprechend auch nicht, dass keine Garantie besteht, dass die angebotenen Inhalte tatsächlich korrekt und vollständig sind(vgl. Kolbitsch u. Marer [31]).

Ein anderes weit verbreitetes Mittel zur Einbeziehung von Nutzern im Web 2.0 sind o genannte Tags oder Schlagwörter (vgl. Golder u. Huberman[21]). Viele Internet-Seiten bieten ihren Besuchern heute die Möglichkeit, die angebotenen Inhalte mit Schlüsselwörtern zu versehen, die nach deren Ansicht das betreffende Objekt beschreiben. Dabei gibt es keine

Vorgaben wie diese Tags auszusehen haben, es handelt sich ähnlich wie bei Wikis um ein Selbst-organisierendes System, dass unter der Annahme funktioniert, dass, wenn genügend Benutzer Schlüsselwörter eintragen, auf lange Sicht eine korrekte und vollständige Beschreibung entsteht. Derartige Tagging-Systeme gelten als mögliche Alternative zu Ontologien zur semantischen Beschreibung von Web-Inhalten, es gibt allerdings einige Probleme und Unterschiede. Bei Tags gibt es keine Hierarchie, Suchfunktionen die auf Tags zugreifen entsprechen einer einfachen Schlüsselwort-Suche. Die Semantik ergibt sich aus der Verwendung von Schlüsselwörtern, die eben diese beschreiben, und aus Überschneidungen mit den Beschreibungen anderer Objekte. Mehrdeutige Schlüsselwörter stellen kein Problem dar, solange sie bei einer Suchanfrage in Kombination mit anderen Begriffen verwendet werden, aus denen der Kontext hervorgeht. Schwieriger wird es allerdings bei Synonymen oder verschiedenen Formen eines Wortes (z.B. Singular und Plural). Diese müssen jeweils als eigene Tags angegeben werden, während bei Ontologien die Verbindung über ein gemeinsames Konzept hergestellt werden kann. Des Weiteren ist die Granularität der verwendeten Begriffe von Bedeutung. Dies gilt sowohl für die Tags, als auch für die verwendeten Suchbegriffe, so könnte z.B. eine Suche nach „Programmierung“ keine Objekte finden die z.B. mit „Java“ oder „Perl“ beschrieben sind, da der hierarchische Zusammenhang zwischen diesen Begriffen nicht abgebildet wird. Nach der zugrunde liegenden Theorie der selbstorganisierenden Systeme nehmen diese Probleme mit zunehmender Benutzer- und Tag-Anzahl ab, sie zeigen daher, wie wichtig eine ausreichend große Zahl an Benutzern und Informationen ist, bevor Community-generierte Inhalte eine zuverlässige Datenbasis für eine Suchfunktion bieten.

Abbildung 2.3: Tag Cloud auf Seekda.com[53]

Tags können nicht nur als Grundlage für eine schlüsselwortbasierte Suche dienen, sondern auch zur Auswahl ähnlicher Objekte. So bieten viele Webseiten die Möglichkeit, die zu einem beschriebenen Objekt angezeigten Tags direkt anzuklicken, um so Objekte zu erhalten, die

über dieselben Tags verfügen. Eine weitere mögliche Verwendung von Tags sind Tag Clouds (s. Abbildung 2.3). Dabei handelt es sich um eine Sammlung der am häufigsten verwendeten bzw. gesuchten Tags, wobei die Verwendungshäufigkeit durch die Schriftgröße der Tags dargestellt wird. Tag Clouds bieten die Möglichkeit schnell und einfach auf die am häufigsten gesuchten Inhalte einer Webseite zuzugreifen, ohne eine Suchanfrage formulieren zu müssen.

2.4 Informationsgewinnung aus Texten

Nicht alle Wörter, aus denen natürlichsprachliche Texte wie z.B. Wiki-Einträge oder Textdokumente bestehen, sind für deren Semantik von Bedeutung. Bevor ein Programm in der Lage ist, derartige Texte zur Bestimmung der Semantik zu verwenden, müssen sie erst bereinigt und auf die Nutzung durch eine Anwendung vorbereitet werden. Die folgenden Abschnitte beschreiben Verfahren um relevante Schlüsselwörter aus natürlichsprachlichen Texten zu extrahieren.

2.4.1 Vorbereitung von Texten

Der erste Schritt beim Auswerten von Textdokumenten ist häufig die Entfernung von Stoppwörtern (vgl. Kuck et al.[33], Reichling et al.[50]) Dabei handelt es sich um Wörter die häufig benutzt werden, aber keine Bedeutung für den Inhalt des Textes haben[3]. Ohne diese Wörter zu entfernen, könnte z.B. eine Suchmaschine annehmen, dass zwei Dokumente die gleiche Bedeutung haben, weil in beiden häufig das Wort „der“ verwendet wird. Gewöhnlich wird eine Stoppwort-Liste verwendet, die alle Wörter enthält, die bei der weiteren Auswertung des Textes ignoriert werden sollen. Neben Stoppwörtern ist das Vorkommen verschiedener morphologischer Variationen von Wörtern ein Problem bei der Textanalyse, so können z.B. Singular und Plural eines Wortes als zwei verschiedene Wörter erkannt werden. Eine Lösung für dieses Problem ist „Stemming“, die Reduktion von Wörtern auf eine Grundform, die alle Varianten eines Wortes gemein haben. Caumanns [14] stellt einen Algorithmus zur Durchführung dieser Reduktion bei deutschsprachige Texte vor. Dazu werden Endungen wie „-er“, „-en“, „-s“ oder „-te“ abgeschnitten, Umlaute und andere Besonderheiten der deutschen Sprache, die sich auf verschiedene Formen von Wörtern auswirken werden, ersetzt. Die Art des Wortes (z.B. Verb oder Nomen) sowie der Kontext werden nicht betrachtet, was dazu führen kann, dass zu viel abgeschnitten wird und aus der reduzierten Form das tatsächliche Wort nicht mehr erkennbar ist. Dies ist aber solange kein

3 In der deutschen Sprache z.B. „der“, „die“, „das“, „und“, „oder“, „weil“, „ein“,...

Problem, wie alle Formen des Wortes auf die selbe Grundform reduziert werden. Fehler lassen sich bei diesem Verfahren nicht völlig vermeiden. Man unterscheidet zwischen zwei Arten von Fehlern. Der erste tritt auf, wenn zwei Formen desselben Wortes nach dem Stemming nicht dieselbe reduzierte Form bilden („Understemming“), die zweite wenn verschiedene Wörter über dieselbe Grundform verfügen („Overstemming“). „Understemming“ tritt beispielsweise bei Wörtern griechischen oder lateinischen Ursprungs, mit ungewöhnlichen Mehrzahlen auf (z.B. „Drama“/“Dramen“, „Minimum“/“Minima“[14]). Verhindern lassen sich derartige Fehler durch die Verwendung von Wortlisten mit Wörtern, deren morphologische Variationen gegen die üblichen Regeln der deutschen Sprache verstoßen. „Overstemming“ lässt sich z.B. bei den Wörtern „Buch“, „Büchse“, „Buchse“ und „Büchner“ beobachten, die nach Entfernung von Endungen und Umlauten alle dieselbe Form „Buch“ bilden. Das Ergebnis der Studie zum vorgestellten Algorithmus war, dass die Fehlerquote, falls er nur auf Nomen angewendet wird bei unter 1% liegt. Werden alle Wörter berücksichtigt steigt diese auf bis zu 15%,lässt sich jedoch mit Hilfe von Listen mit irregulären Wörtern auf unter 5% senken, was für die meisten Anwendungen als akzeptabel gilt (vgl. Caumanns[14]).

Stemming-Algorithmen unterscheiden sich je nach Sprache, im englischen ist z.B. der PorterStemming-Algorithmus(vgl. van Rijsbergen et al.[61]) verbreitet. Das grundsätzliche Verfahren ist bei allen Stemming-Algorithmen dasselbe, es geschieht lediglich eine Anpassung an sprachspezifische Regeln.

Ein Beispiel für die Verwendung von Stoppwort-Filtern und Stemming-Algorithmen findet sich im bereits vorgestellten Service Discovery System von Kuck et al.[33]. Dort werden Operations-- und Parameternamen aus WSDL-Dateien in ihre Bestandteile zerlegt (z.B. „getCanadaWeather“ in „get“, „Canada“ und „Weather“). Aus den so gewonnenen Schlüsselwörtern werden die Stoppwörter entfernt (z.B. „get“) und sie werden über einen Stemming-Algorithmus normalisiert, um dann für die Beschreibung der Funktionalität des Dienstes im Service-Profil verwendet zu werden.

2.4.2 Gewichtung von Schlüsselwörtern

Ziel der Textanalyse ist es, aus vorliegenden Dokumenten Schlüsselwörter zu extrahieren um diese dann mit einer Suchanfrage zu vergleichen. Sowohl Schlüsselwörter als auch Suchbegriffe werden als Vektoren dargestellt, die dann miteinander verglichen werden. Zur Berechnung der Ähnlichkeit wird im einfachsten Fall die Übereinstimmung zwischen den Elementen der beiden Vektoren berechnet werden. Da allerdings nicht jeder Begriff, der in einem Dokument vorkommt, für den Inhalt von gleicher Bedeutung ist, kann es sinnvoll sein

eine Gewichtung für Schlüsselwörter einzuführen. Ein weit verbreitetes Verfahren ist TF-IDF (term frequency - inverse document frequency), das nach G. Salton und C. Buckley [52] drei Faktoren zur Berechnung der Wortgewichtung verwendet. Zuerst wird die Termfrequenz benötigt, die aussagt, wie oft ein Begriff im Verhältnis zu anderen Begriffen in einem Dokument vorkommt. Wörter, die am häufigsten in einem Text vorkommen werden als am wichtigsten für dessen Inhalt betrachtet. Die Termfrequenz sorgt dafür, dass relevante Wörter erkannt werden, allerdings werden irrelevante nicht ausgeschlossen. So können z.B. allgemeine Begriffe in vielen Dokumenten auftauchen und somit zu vielen, ungenauen Suchergebnissen führen. Zur Erhöhung der Präzision wird deshalb der zweite Faktor, die inverse Dokumenthäufigkeit eingeführt. Diese gibt an, wie häufig ein Schlüsselwort in der gesamten Dokumentensammlung vorkommt. Die Annahme ist, dass je häufiger ein Wort in unterschiedlichen Dokumenten vorkommt, desto weniger ist es als Auswahlkriterium zwischen verschiedenen möglichen Suchergebnissen geeignet und sollte entsprechend niedriger gewichtet werden. Falls die betrachteten Dokumente in ihrem Umfang stark variieren, kann es sinnvoll sein, als drittes Element noch einen Normalisierungsfaktor zu verwenden. In umfangreichen Dokumenten können Begriffe häufiger vorkommen als in kleinen Dokumenten, entsprechend werden unabhängig von der tatsächlichen Relevanz des Inhalts Dokumente mit vielen Wörtern gegenüber kurzen Dokumenten als Suchergebnisse bevorzugt. Die Aufgabe des Normalisierungsfaktors ist es, die Längen der Schlüsselwortvektoren aller Dokumente anzugleichen, um die Benachteiligung kleiner Dokumente zu vermeiden. Eine Verwendung von TF-IDF in den bereits vorgestellten Beispielen findet sich bei Kawamura et al.([29]), deren Service Discovery System neben den semantischen Beschreibungen der Dienste einen Textfilter verwendet, der per TF-IDF Informationen aus menschenlesbaren Beschreibungen extrahiert, um die Suchergebnisse zu verbessern.

TF-IDF hilft bei er Verbesserung von einfachen Schlüsselwortsuchen, das bedeutet, dass nur Ergebnisse geliefert werden, wenn ein Schlüsselwort wörtlich in einem Dokument vorkommt. Mehrdeutige Begriffe, Synonyme und andere semantische Zusammenhänge können nicht erkannt werden. Eine Möglichkeit, die Semantik in eine Suche einzubeziehen bieten Ontologien (s. Kapitel 2.2.1), deren Aufbau und Wartung jedoch aufwändig ist. Eine Alternative stellt LSI („Latent Semantic Indexing“, vgl. Deerwester et al.[15]) dar. LSI ist ein mathematisches Verfahren, bei dem anhand einer Term-Dokument-Matrix, in der alle Begriffe jedes betrachteten Dokuments verzeichnet sind, die semantische Ähnlichkeit zwischen den Dokumenten berechnet wird. Zur Verringerung der Komplexität werden zuerst unwichtige (z.B. mit TF-IDF niedriger gewichtete) Begriffe entfernt, und Wörter mit ähnlicher Bedeutung zu Konzepten zusammengefasst. Diese reduzierte Form des Dokumentinhalts kann nun zur Berechnung der Ähnlichkeit von Such-Vektoren zu Dokumenten genutzt

werden, die aufgrund der Verdichtung der Wörter auch zu Treffern führen kann, wenn keine wörtliche Übereinstimmung vorliegt. Ein Beispiel für eine Anwendung, die unter anderem TF-IDF und LSI verwendet ist, das ExpertFinding-System [50], dessen Ziel es ist, Benutzer bei der Suche nach Experten in einem bestimmten Fachgebiet zu unterstützen und ihr eigenes Wissen anderen Nutzern zur Verfügung zu stellen. Um festzustellen, in welchem Fachgebiet sich ein Benutzer auskennt, werden Dokumente, die sich auf seinem Computer befinden analysiert um aus ihnen nach der Entfernung von Stoppwörtern ein Schlüsselwort-Profil zu erzeugen. Zum Auffinden von Experten werden Suchanfragen der Benutzer und die Schlüsselwortprofile anderer Nutzer über eine einfache Schlüsselwortsuche, eine TF-IDF-basierte Suche, sowie eine auf LSI aufbauende Suchfunktion miteinander verglichen.

2.5 Ähnlichkeitsbestimmung

In den vorherigen Kapiteln wurde mehrfach der Begriff der Ähnlichkeit zwischen Konzepten, Begriffen oder Dokumenten verwendet und verschiedene Möglichkeiten vorgestellt, diese zu berechnen. Im Folgenden wird die Qualität derartiger Verfahren behandelt und ein weiteres, auf Ontologien basierendes Verfahren zur Berechnung von semantischer Ähnlichkeit vorgestellt.

2.5.1 Qualität von Algorithmen zur Ähnlichkeitsberechnung

Ontologien können als Graphen verstanden werden, in denen Verbindungen zwischen Objekten zur Berechnung der Ähnlichkeit verwendet werden können. Objekte, die in der Ontologie näher beieinander liegen, also durch einen kürzeren Pfad verbunden sind, gelten als ähnlicher (vgl. Bianchini et al.[11]). Beim TF-IDF-Verfahren werden gewichtete Schlüsselwortvektoren miteinander verglichen um Ähnlichkeiten zwischen Dokumenten, bzw. zwischen Dokumenten und Suchbegriffen zu berechnen. Ziel derartiger Verfahren ist es, die Ähnlichkeit so einzuschätzen, wie es ein Mensch tun würde, der die Ähnlichkeit derselben Objekte beurteilt. Um die Qualität von Ähnlichkeitsberechnungen zu bestimmen, beschäftigt sich die Studie von Bernstein et al.[10] daher mit dem Vergleich von Ähnlichkeitseinschätzungen von Versuchspersonen, und verschiedenen Verfahren zur automatischen Berechnung der Ähnlichkeit.

Für diese Studie wurden aus einer Ontologie[4] Paare von Objekten entnommen, deren Ähnlichkeit von einer Gruppe von Testpersonen beurteilt wurde. Daraufhin wurde dieselbe Aufgabe mit Hilfe verschiedener Algorithmen zur Ähnlichkeitsberechnung ausgeführt,

4 dem MIT-Process-Handbook (http://ccs.mit.edu/ph/)

darunter die Distanz der Objekte innerhalb der Ontologie und ein TF-IDF-basiertes Verfahren, das den Konzepten zugeordnete Beschreibungstexte auswertete. Ziel war es festzustellen, ob sich die Ergebnisse der Menschen und automatischen Verfahren voneinander unterscheiden, es zeigte sich jedoch, dass es schon zwischen den Beurteilungen der verschiedenen Testpersonen erhebliche Unterschiede gab. Gleichzeitig wichen die Ergebnisse der Algorithmen nicht weiter von denen der Testpersonen ab, als die Ergebnisse der Testpersonen untereinander, es hätte sich demnach auch um die subjektive Beurteilung einer weiteren Testperson handeln können. Die Teilnehmer wurden in Gruppen mit ähnlichen Beurteilungen eingeteilt und es zeigte sich, dass die Ergebnisse des TF-IDF-Verfahrens am konsistentesten einer dieser Gruppen fest zugeordnet werden konnte. Die Schlussfolgerung aus der Studie war daher, dass es keine automatische Berechnung der Ähnlichkeit geben kann, die der subjektiven Beurteilung von Ähnlichkeit jedes Menschen entspricht, da nicht jeder Mensch die Ähnlichkeit von Objekten auf dieselbe Weise einschätzt. Falls die Ähnlichkeitsbestimmung Teil eines Prozesses ist, an dem menschliche Benutzer beteiligt sind, wird eine Personalisierung der Berechnung vorgeschlagen, bei der an das Ähnlichkeitsempfinden des jeweiligen Benutzers angepasste Algorithmen verwendet werden.

2.5.2 Ähnlichkeitsgraphen

Ein weiteres Verfahren zur Berechnung von Ähnlichkeiten sind die, auf Ontologien aufbauenden Ähnlichkeitsgraphen (vgl. Andreasen, Knappe, et al.[30],[6]). Aus einer Ontologie wird ein gerichteter, azyklischen Teilgraph extrahiert, der die zu vergleichenden Konzepte enthält. Abbildung 2.4 zeigt einen solchen, aus einer Ontologie entnommenen Teilgraphen, der Zusammenhänge zwischen verschiedenen Tierarten beschreibt.

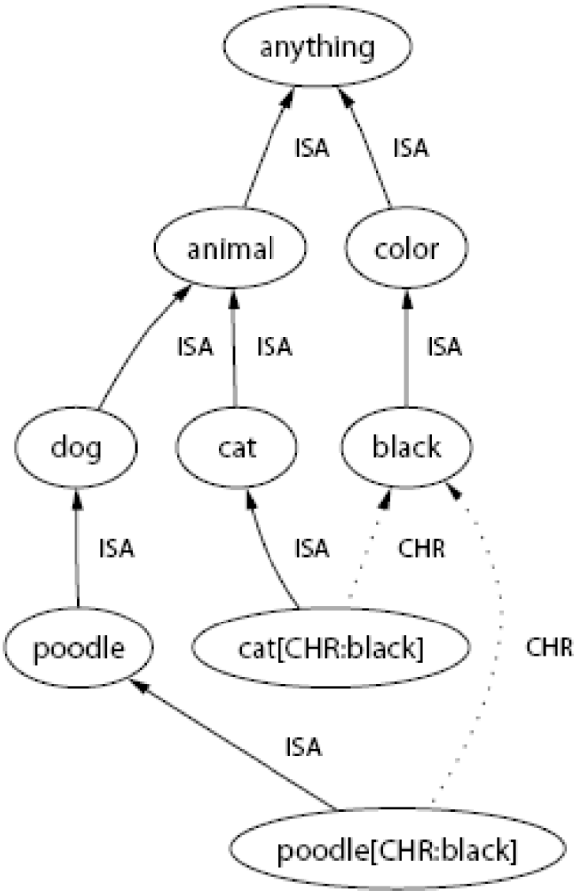

Abbildung 2.4: Beispiel für einen Ähnlichkeitsgraphen [6]

Im Beispiel gibt es zwei Arten von Beziehungen, „Ist ein“ („ISA“) und „charakterisiert durch“(„CHR“). So kann man aus der Abbildung ableiten, dass ein Pudel ein Hund ist und es sich bei Hunden um Tiere handelt. Ein schwarzer Pudel ist ebenfalls ein Pudel, der jedoch durch die Eigenschaft „schwarz“ genauer beschrieben wird. Im Graphen ist ebenfalls erkennbar, dass es sich bei „schwarz“ um eine Farbe handelt. Die Berechnung der Ähnlichkeit erfolgt über den Vergleich der von beiden betrachteten Knoten gemeinsam erreichbaren Knoten. Wird, wie zuvor festgestellt, die Länge des kürzesten Weges zwischen zwei Konzepten innerhalb der Ontologie als Kriterium für die Ähnlichkeit verwendet, so wird

die Existenz von alternativen Pfaden, also zusätzlichen semantischen Zusammenhängen zwischen den Konzepten vernachlässigt. Des Weiteren werden genau wie bei textbasierten Verfahren wie TF-IDF oder LSI hierarchische Zusammenhänge der Objekte untereinander nicht betrachtet. Die Grundform, der zur Berechnung der Ähnlichkeit verwendeten Formel lautet:

$$\mathrm{sim}(x,y)=\frac{|\alpha(x)\cap\alpha(y)|}{|\alpha(x)|}$$ (vgl. Andreasen et al. [7])

x und *y* sind die betrachteten Knoten, $|\alpha(x)|$ bzw. $|\alpha(y)|$ ist die Anzahl aller von *x* bzw. *y* entlang der Kantenrichtung erreichbaren Knoten. Die Formel berechnet die Ähnlichkeit von *x* zu *y* als den Anteil der von *x* zu erreichenden Knoten, die auch von *y* aus erreichbar sind. Dabei kann *sim(x,y)* ungleich *sim(y,x)* sein, da sich die Anzahl der insgesamt von *x* und *y* aus erreichbaren Knoten unterscheiden kann. Dies lässt sich verdeutlichen, wenn man in Abbildung 2.5 die Ähnlichkeit zwischen den Knoten D und E betrachtet.

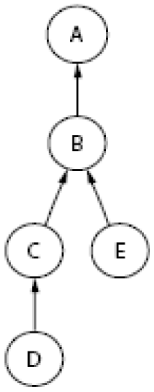

Abbildung 2.5:Einfacher Ähnlichkeitsgraph [6]

Von *D* aus sind die Knoten *D*, *C*, *B* und *A*, also insgesamt 4 Knoten in Kantenrichtung erreichbar. Bei E sind es nur drei Knoten, nämlich *E*, *B* und *A*. Die beiden Knoten *B* und *A* sind sowohl von *D*, als auch von *E* erreichbar. Setzt man diese Zahlen in die oben vorgestellte Formel ein, so ergibt sich:

$$|\alpha(D)|=4$$

$$|\alpha(E)|=3$$

$$|\alpha(D)\cap\alpha(E)|=2$$

$$\mathrm{sim}(D,E)=\frac{2}{4}=\frac{1}{2}$$

$$\mathrm{sim}(E, D) = \frac{2}{3}$$

Die Asymmetrie ergibt sich aus der Hierarchie zwischen den betrachteten Objekten. So stehen „Ist ein“-Beziehungen für eine Generalisierung, in Abbildung 2.4 ist z.B. „animal“ eine Generalisierung von „dog“, umgekehrt ist „dog“ eine Spezialisierung von „animal“. Um die Auswirkungen der Hierarchie auf die Ähnlichkeit einzubeziehen, werden in [6] drei Eigenschaften für derartige Hierarchien formuliert.

- *Generalisierungskosten:* Eine Generalisierung sollte sich negativer auf die Ähnlichkeit auswirken, als eine Spezialisierung. Die intuitive Begründung für diese Eigenschaft ist, dass z.B. jeder Hund ein Tier ist, aber nicht jedes Tier ein Hund. Entsprechend sollte eine Ähnlichkeit umso höher gewichtet werden, je spezieller ein Konzept ist. Diese Eigenschaft wird von der oben genannten Formel erfüllt, und findet sich in der bereits betrachteten Asymmetrie zwischen *sim(D,E)* und sim*(E,D)* in Abbildung 2.5 wieder. Die Distanz zwischen den beiden Knoten im Graphen ist gleich, egal von welchem Knoten aus sie betrachtet wird. Da *D* jedoch ein speziellerer Knoten ist als *E* gilt: *sim(D,E)* < *sim(E,D)*. Dabei ist zu beachten, dass dieses Ergebnis nur dann stimmt, wenn ansonsten alle Bedingungen gleich sind. Die Eigenschaft besagt nicht, dass wenn E ein speziellerer Knoten als D ist in jedem beliebigen Graphen sim(D,E) kleiner sein muss als sim(E,D), sondern nur, dass die Generalisierung zusätzliche Kosten verursacht, so dass aus Sicht von Knoten D eine größere Übereinstimmung mit E vorliegen müsste um dasselbe Ergebnis zu erreichen. Diese Einschränkung gilt auch für die beiden folgenden Eigenschaften.

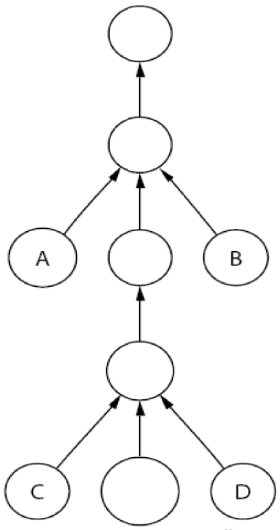

Abbildung 2.6: Spezifitätskosten in Ähnlichkeitsgraphen

- *Spezifitätskosten:* Ähnlichkeiten zwischen generellen Knoten sollten geringer gewertet

werden, als zwischen spezielleren Knoten. Auch dies lässt sich wieder intuitiv begründen, da eine Ähnlichkeit zwischen abstrakten Begriffen weniger Bedeutung für einen möglichen Benutzer hat, als eine auf den ersten Blick genauso große Ähnlichkeit zwischen zwei speziellen Begriffen. Abbildung 2.6 zeigt ein Beispiel für diese Eigenschaft. Die Knoten A und B sind allgemeiner als die Knoten *C* und *D*, entsprechend müsste *sim(A,B)* kleiner sein als *sim(C,D)*. Die bisher verwendete Formel erfüllt auch diese Anforderung. Von *A* und *B* sind jeweils 3 Knoten aus erreichbar, davon zwei von beiden Knoten aus, *sim(A,B)* ist also 2/3. von den Knoten *C* und *D* sind jeweils 5 Knoten erreichbar, davon können 4 von beiden Knoten erreicht werden, was für *sim(C,D)* einen Wert von 4/5 ergibt.

- *Spezialisierungskosten:* Diese Eigenschaft besagt, dass eine weitere Spezialisierung die Ähnlichkeit senken sollte, dass also die Ähnlichkeit zu einem Knoten auf derselben Ebene höher ist, als zu einem spezielleren Knoten, auch wenn die gemeinsam erreichbaren Knoten gleich bleiben. Diese Eigenschaft wird von der bisher benutzten Formel nicht erfüllt, so gilt z.B. in Abbildung 2.5 *sim(E,C)=sim(E,D)*. C und D sind zwar unterschiedlich weit von E entfernt, allerdings stimmt die Anzahl der gemeinsam erreichbaren Knoten überein, so dass dieselbe Ähnlichkeit berechnet wird.

Da die Generalisierungskosten und die damit verbundene Asymmetrie der Ähnlichkeit nicht aufgegeben werden darf, wird die Formel wie folgt verändert:

$$\mathrm{sim}(x,y)=p\frac{|\alpha(x)\cap\alpha(y)|}{|\alpha(x)|}+(1-p)\frac{|\alpha(x)\cap\alpha(y)|}{|\alpha(y)|}$$ (vgl. Andreasen et al. [7])

Der rechte Bruch ist die Gegenrichtung der Rechnung, die zur Umsetzung der dritten Eigenschaft dient. Diese wird mit dem Faktor *p* gewichtet, dessen Wert zwischen *0* und *1* liegt. Ist *p=1*, so ist der rechte Bruch gleich *0*, somit ergibt sich die bereits bekannte Formel ohne Berücksichtigung der Spezialisierungskosten. Der Einfluss der Spezialisierungskosten steigt, je niedriger das gewählte *p* ist. Andreasen et al. [7] verwenden in ihren Berechnungen *p=4/5*. Setzt man diesen Faktor und die Werte des Beispiels aus Abbildung 2.5 in die neue Formel ein, so ergibt sich für sim(E,C) und sim(E,D):

$$|\alpha(E)|=3$$

$$|\alpha(D)|=4$$

$$|\alpha(C)|=3$$

$$|\alpha(E)\cap\alpha(C)|=2$$

$$|\alpha(E) \cap \alpha(D)| = 2$$

$$\text{sim}(E, C) = \frac{4}{5} * \frac{2}{3} + \frac{1}{5} * \frac{2}{3} = \frac{2}{3} \approx 0{,}67$$

$$\text{sim}(E, D) = \frac{4}{5} * \frac{2}{3} + \frac{1}{5} * \frac{2}{4} = \frac{19}{13} \approx 0{,}64$$

Die erweiterte Formel errechnet einen anhand des Faktors p gewichteten Durchschnitt zwischen der betrachteten Ähnlichkeit und ihrer Gegenrichtung, also der Ähnlichkeit vom Ziel zum Ausgangsknoten. Für sim(E,C) ändert sich dadurch am Ergebnis nichts, da die beiden Knoten auf derselben Ebene im Graphen liegen und die Ähnlichkeit daher in beide Richtungen gleich ist. Da jedoch aufgrund der Generalisierungskosten gilt, dass sim(E,D) größer ist als sim(D,E), ergibt sich nach Einbeziehung der Gegenrichtung ein geringerer Wert für sim(E,D), somit ist die Anforderung sim(E,C)>sim(E,D) erfüllt.

2.6 Personalisierung

Ein wichtiger Teil der Fragestellung dieser Arbeit ist neben der Suchfunktion auch die Abgabe von personalisierten Empfehlungen, basierend auf Benutzerprofilen. Im Folgenden werden verschiedene Ansätze vorgestellt, wie dies erreicht werden kann.

2.6.1 Recommender Systeme

Heutzutage nutzen viele Internet-Seiten Empfehlungssysteme, um ihren Nutzern bzw. Kunden auf ihre persönlichen Bedürfnisse angepasste Informationen zu bieten und Angebote zu machen. Ein bekanntes Beispiel ist der Online-Versandhandel Amazon.de[5]. Dort bekommen Besucher auf Basis der von ihnen betrachteten oder in der Vergangenheit gekauften Artikel weitere Produkte angezeigt, die für sie von Interesse sein könnten (s. Abb. 2.7).

Allgemein besteht die Aufgabe eines Recommender-Systems darin, festzustellen wie nützlich ein Objekt für einen Benutzer ist, indem Benutzerprofile und andere Informationen über den Benutzer und die zu empfehlenden Objekte zur Berechnung einer Punktzahl ausgewertet werden. Diese Punktzahl dient als Vorhersage, wie interessant ein Objekt voraussichtlich für einen Benutzer ist. Der Inhalt des Benutzerprofils wird entweder explizit vom Nutzer eingegeben oder automatisch durch Beobachtung des Nutzungsverhaltens erzeugt.

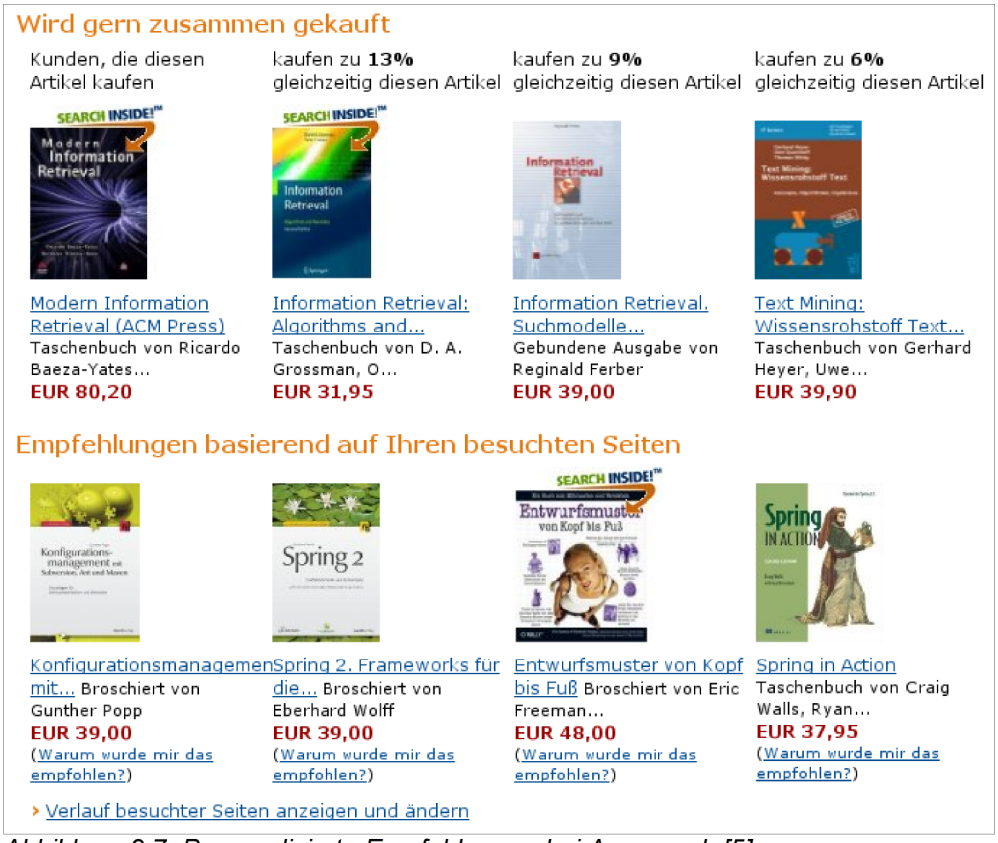

Abbildung 2.7: Personalisierte Empfehlungen bei Amazon.de[5]

Nach Mobasher[39] können Recommender-Systeme in drei Kategorien unterteilt werden. Der erste Typ sind wissensbasierte Recommender. Diese benutzen Informationen über betrachteten Objekte und Benutzer. Anhand von festen Regeln werden diese einander zugeordnet. Die verwendeten Informationen und auch die Regeln, bestimmen die Kriterien für Empfehlungen und letztendlich auch deren Qualität und werden abhängig von der Domäne des jeweiligen Recommender-Systems festgelegt.

Inhaltsbasierte Empfehlungssysteme sammeln im Benutzerprofil Informationen über Objekte, an denen der Benutzer in der Vergangenheit interessiert war, also z.B. im Falle von Amazon.de betrachtete Produkte oder getätigte Käufe. Aus den Informationen über diese Objekte, also für gewöhnlich Beschreibungen in Textform werden verschiedene Eigenschaften extrahiert und mit anderen Objekten verglichen, an denen bisher nicht explizit Interesse geäußert wurde. Anhand der Ähnlichkeit mit aus dem Benutzerprofil bekannten Objekten, wird eine Punktzahl berechnet, die eine Vorhersage des Grades an Interesse für den Benutzer darstellt. Ein Haupt-Kritikpunkt an inhaltsbasierten Systemen ist die Spezialisierung auf Objekte die in der Vergangenheit von Interesse waren, auch wenn der betroffene Benutzer bei seiner nächsten Nutzung des Systems möglicherweise ein völlig anderes Ziel verfolgt.

Dieses Problem soll durch die dritte Form von Recommender-Systemen, das „Collaborative-Filtering“ behoben werden. Anstatt Objekte nach Ähnlichkeit zu sortieren, werden hier die Benutzer anhand von ihnen zu verschiedenen Objekten abgegebenen Bewertungen verglichen. Zwei Benutzer sind sich umso ähnlicher, je näher die zu den selben Objekten abgegebenen Bewertungen durchschnittlich sind. Das voraussichtliche Interesse eines Nutzers an einem Objekt berechnet sich aus den Bewertungen die ähnliche Benutzer für dieses Objekt abgegeben haben. Auch das „Collaborative-Filtering“ hat Schwächen, insbesondere im Bezug auf die Skalierbarkeit. Zum Auffinden ähnlicher Benutzer muss jeder Benutzer mit jedem andern verglichen werden, was mit zunehmender Benutzerzahl einen großen Zeitaufwand bedeuten kann. Des Weiteren kann es schwierig werden, ausreichende Übereinstimmungen zwischen Nutzern zu finden, wenn es sehr viele Objekte gibt, auf die sich die Bewertungen weniger Nutzer aufteilen. Ein Beispiel eines auf diesem Prinzip basierenden Empfehlungssystems ist Movielens[40], ein Projekt der Universität von Minnesota. Benutzer haben hier die Möglichkeit Filme zu bewerten und bekommen zu noch nicht bewerteten Filmen eine Vorhersage darüber, wie gut ihnen diese Filme voraussichtlich gefallen werden. Diese basieren auf den abgegebenen Bewertungen für diesen Film von anderen Nutzern mit einem ähnlichen Filmgeschmack. Die Berechnung der Ähnlichkeit zwischen Benutzern geschieht bei Movielens nicht in Echtzeit, sondern offline in bestimmten Abständen. Dadurch wird einerseits das Problem der Skalierbarkeit abgeschwächt, andererseits beeinflussen abgegebene Bewertungen die Vorhersagen erst zeitversetzt.

2.6.2 Relevanz-Feedback

Eine weitere Möglichkeit der Personalisierung ist Relevanz-Feedback. Shanfeng et al.[54] stellen ein System vor, das Relevanz-Feedback durch Benutzer verwendet, um die Ergebnisse einer Meta-Suchmaschine, die Ergebnisse mehrerer anderer Suchmaschinen zusammenfasst zu personalisieren. Laut dieser Arbeit liegt eine der Hauptschwierigkeiten beim Entwurf von Suchsystemen darin, dass durchschnittliche Suchanfragen nur aus wenigen Wörtern bestehen und Benutzer allgemein nicht gut darin sind, ihre Wünsche präzise zu formulieren. Nachdem aber z.B. im vorliegenden Fall eine Webseite gefunden und gelesen wurde, weiß der Benutzer, ob die gefundenen Informationen für seine Suchanfrage relevant sind und kann Suchergebnisse entsprechend als relevant oder nicht relevant markieren. Diese Bewertung wirkt sich nicht nur auf das bewertete Suchergebnis selbst aus, sondern auch auf ähnliche Ergebnisse, also im Falle von Web-Seiten andere Seiten mit ähnlichem Inhalt. Die ungenaue Suchanfrage rückt in den Hintergrund, da das Suchsystem nun anhand des abgegebenen Feedbacks weiß, welche Informationen tatsächlich gesucht, bzw. nicht gesucht wurden, und bewertet ähnliche Inhalte entsprechend höher bzw.

niedriger. Im Anwendungsbeispiel von Shanfeng et al.[54] wird das Feedback zur Verbesserung der Suchergebnisse verwendet, und nicht um die Ergebnisse an die Bedürfnisse unterschiedlicher Benutzer anzupassen. Eine Personalisierung durch Feedback wird allerdings in der Arbeit von Kuck et al.[33] vorgestellt. Dort werden, wenn Benutzer positives Feedback abgeben, Begriffe aus deren Benutzerprofil in das Serviceprofil übernommen, so dass die Relevanz des Dienstes für Benutzer mit Übereinstimmungen im Profil in Zukunft höher eingeschätzt wird.

2.6.3 Beurteilung

In diesem Abschnitt wurden eine Reihe von Techniken vorgestellt, Benutzern personalisierte Inhalte bereitzustellen, die zum Teil auf ein Service Discovery System übertragbar sind. Eine wichtige Feststellung ist jedoch, dass der Benutzer einer Software zur Orchestrierung von Web Services das Programm nutzt um eine bestimmte Aufgabe zu erfüllen. Ist dies einmal geschehen, wird er es kaum nutzen um eine weitere Komposition aus Diensten mit derselben Funktionalität zu erstellen. Damit scheidet der inhaltsbasierte Ansatz für Recommender-Systeme, der auf dem vergangenen Verhalten eines Benutzers basiert aus, da es nicht darum geht, basierend auf dem Benutzerprofil automatische Empfehlungen auszusprechen, sondern die Ergebnisse einer Suche nach vom Nutzer festgelegten Suchbegriffen zu personalisieren. Vielversprechender erscheint hier das „Collaborative-Filtering“, dass allerdings eine große Benutzerzahl sowie eine ausreichende Zahl an Bewertungen von jedem Benutzer benötigt, da zur aussagekräftigen Berechnung von Ähnlichkeiten zwischen Benutzern genügend Überschneidungen vorhanden sein müssen. Relevanz-Feedback erscheint als gute Möglichkeit zur Personalisierung von Suchergebnissen. Die Hauptschwäche liegt in der notwendigen, expliziten Bewertung der Suchergebnisse durch den Benutzer.

2.7 Zusammenfassung

Diese Arbeit versucht die folgende Frage zu klären:

Wie können community-generierte Inhalte und Benutzerprofile in einen Suchalgorithmus für Webservices integriert werden, um die Ergebnisqualität zu verbessern?

Bevor darauf eine Antwort gefunden werden kann, wurde in diesem Kapitel zuerst beschrieben, welche Schwächen vorhandene Suchsysteme haben und warum die Entwicklung eines neuen Suchsystems als sinnvoll erscheint.

Ein Hauptproblem liegt bei der Beschreibung der Dienste. Häufig werden Webservices von den Dienstanbietern gar nicht oder nur unzureichend dokumentiert, und selbst wenn eine Beschreibung vorliegt weicht der verwendete Wortgebrauch häufig von dem der Nutzer ab, unter anderem wegen der unterschiedlichen Perspektiven auf die Verwendung des Dienstes. Die heute vorhandenen Suchsysteme reichen nicht aus um die Mängel der Dokumentation zu beheben. Insbesondere UDDI, der Webservice-Standard zur Dienstsuche, bietet nur eine sehr einfache Suchfunktion, die nach direkten, syntaktischen Übereinstimmungen zwischen Suchbegriffen und Dienstbeschreibungen sucht, was aufgrund von ungeeigneten Suchbeschreibungen häufig keine Ergebnisse liefert. Als mögliche Lösung für dieses Problem wurden Semantic Web Services vorgestellt, die Dienstbeschreibungen um eine semantische Beschreibung ergänzen und damit Lücken in den eigentlichen Dienstbeschreibungen füllen können.

Als nächstes beschäftigte sich das Kapitel mit den Community-generierten Inhalten. Zuerst musste die Motivation zur Verwendung dieser Inhalte geklärt werden. Damit ein Endbenutzer einen Dienst verstehen kann, muss er ihn erstmal verstehen. Um dies zu erreichen, muss die Beschreibung in einer Form geschehen, die der Perspektive des Nutzers auf das zu lösende Problem entspricht was bei Beschreibungen durch den Dienstanbieter nicht unbedingt der Fall ist. Werden die Beschreibungen innerhalb einer Benutzergemeinschaft generiert, so können diese für die einzelnen Benutzer hilfreichere Informationen bieten. Die Vollständigkeit und Richtigkeit von solchen Informationen ergibt sich aus dem Prinzip der selbstorganisierenden Systeme. Dieses besagt, dass wenn ausreichend viele Benutzer ihr Wissen der Gemeinschaft zur Verfügung stellen, langfristig vollständige und korrekte Inhalte entstehen. Ein Beispiel, in dem dieses Prinzip bereits zu beobachten ist, findet sich in Wiki-Systemen im Internet. Da diese Community-generierten Inhalte die Schwächen von Beschreibungen durch Anbieter zum Teil beheben, die ein großes Problem beim Service Discovery darstellen, scheinen sie auch als Datenbasis für ein Suchsystem geeignet.

Des Weiteren wurde der Begriff der Personalisierung erklärt, indem Empfehlungssysteme vorgestellt wurden, die auf Basis von Informationen über Benutzer bestimmte Objekte empfehlen, die dem Benutzerinteresse entsprechen.

Um die Fragestellung der Arbeit zu klären, soll im Folgenden auf Basis der in diesem Kapitel vorgestellten Verfahren ein Webservice-Suchsystem entwickelt werden, dass als Datenbasis Community-generierte Inhalte verwendet und versucht mit Hilfe von Benutzerprofilen die Qualität der Suchergebnisse weiter zu verbessern.

3 Systementwurf

3.1 Anforderungen

Bevor die genaue Funktionsweise des Webservice Suchsystems vorgestellt werden kann, werden in den folgenden Abschnitten die Anforderungen erläutert, die an die Benutzeroberfläche, die verwendeten Suchalgorithmen sowie die Personalisierung der Suchergebnisse gestellt werden.

3.1.1 Benutzeroberfläche

In Deutschland haben im Jahr 2008 ca. 65% der Bevölkerung Zugriff auf das Internet, was einer Steigerung von 5% im Vergleich zum Jahr 2007 entspricht (vgl. (N)Onliner-Atlas 2008[41]). Informationen im World Wide Web sind nicht klar strukturiert und ihre Menge nimmt ständig zu, was Internet-Suchmaschinen wie z.B. Google[22] zu unersetzlichen Werkzeugen zum Auffinden von Informationen macht, mit denen jeder Internetnutzer früher oder später in Kontakt kommt (vgl. Shanfeng et al. [54]).

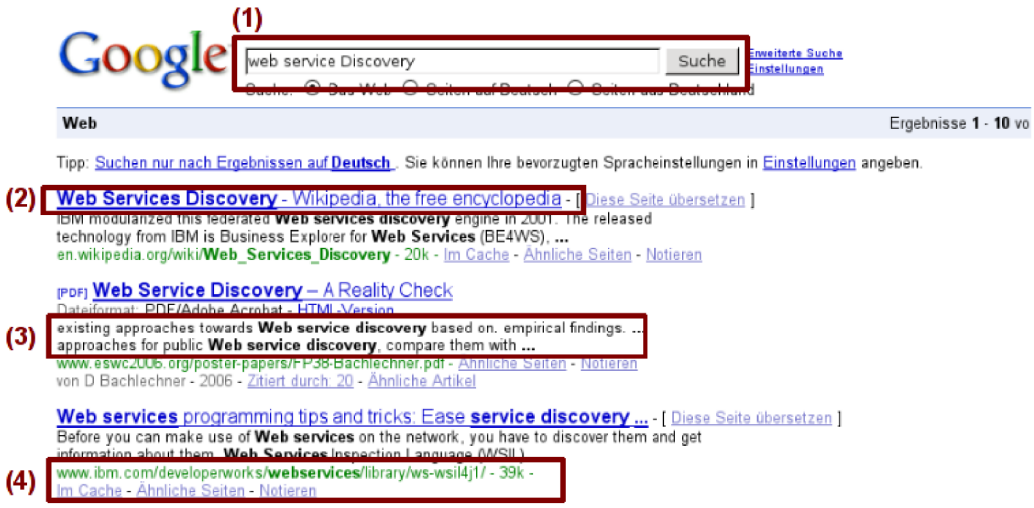

Abbildung 3.1: Beispiel einer Web-Suchmaschine (Google.de[22])

Im World Wide Web gibt es eine Reihe verschiedene Portale zur Web Service Suche (z.B. ProgrammableWeb.com[49], seekda.com[53], die nicht erst seit der Schließung der UBR[37] die erfolgversprechendste Methode zum Auffinden von Webservices darstellen (vgl. Bachlechner et al.[8]). Somit ist anzunehmen, dass ein Großteil der Endbenutzer, für die das Webservice Suchsystem entwickelt wird, mit der Verwendung von Web-Suchmaschinen vertraut ist. Aus diesem Grund dienen derartige Suchmaschinen als Vorlage für die zu

entwickelnde Benutzeroberfläche, um so den Endbenutzern eine vertraute und einfach zu erlernende Bedienung des Webservice Suchsystems zu ermöglichen.

In Abbildung 3.1 wird die Suchseite von Google gezeigt, wobei die wesentlichen, im Folgenden besprochenen Elemente bei den meisten Web-Suchmaschinen übereinstimmen. Die Eingabe von Suchanfragen geschieht über ein Textfeld, in das frei gewählte Suchbegriffe in natürlicher Sprache eingegeben werden können *(1)*. Das Erlernen einer festgelegten Syntax ist nicht zwingend erforderlich. Dieses Prinzip soll in das Suchsystem übernommen werden, um den Endbenutzern das Ausführen von Suchanfragen zu erleichtern.

Bei Web-Suchmaschinen besteht jedes Suchergebnis aus mehreren Zeilen. In der ersten Zeile wird der Titel der gefundenen Seite hervorgehoben *(2)*. Sie dient außerdem als Verknüpfung zum Öffnen der Seite. Die nächsten Zeilen enthalten einen Ausschnitt oder eine Zusammenfassung des Inhalts der gefundenen Internetseite *(3)*. Im Falle von Google werden hier Textstellen angezeigt, in denen die gefundenen Suchbegriffe enthalten sind. Am Ende des Suchergebnisses gibt es je nach Suchmaschine noch Zeilen, die weitere Funktionen, wie z.B. die Suche nach verwandten Seiten ermöglicht *(4)*.

Der Aufbau der Suchergebnisse soll soweit wie möglich übernommen werden. Die erste Zeile soll den Titel des Suchergebnisses enthalten, gefolgt von der Beschreibung des Dienstes bzw. der Operation. Darauf könnten weitere Informationen und Bedienelemente, z.B. zur Abgabe von Relevanz-Feedback und zur Anzeige der vom Suchalgorithmus berechneten Punktzahl folgen. Auch die Verwendung der Suchergebnisse soll einer Web-Suchmaschine entsprechen, d.h. ein einfacher Mausklick auf den Ergebnistitel oder eins der weiteren Bedienelemente soll eine entsprechende Aktion auslösen, also z.B. eine Operation in den Orchestrierungs-Editor übernehmen oder eine Relevanz-Bewertung abgeben.

3.1.2 Berechnung der Suchergebnisse

Die aus Sicht des Endbenutzers entscheidenden Schwächen einer UDDI-basierten Dienstsuche, sind die für die Suche zur Verfügung stehenden, vom Anbieter erstellten Informationen, und die unzureichende, einfache Schlüsselwortsuche die UDDI anbietet (vgl. Kapitel 2.1.1). Wenn Anbieter ihre Dienste überhaupt dokumentieren, dann beschreiben sie die Funktionsweise eines Dienstes. Im Gegensatz dazu ist der Benutzer auf der Suche nach einer Lösung für ein konkretes Problem aus ihrer persönlichen Anwendungsdomäne. Aufgrund unterschiedlicher Hintergründe und Perspektiven können sich die verwendeten Begriffe der beiden Seiten unterscheiden, wodurch die Schlüsselwortsuche, die nach syntaktischen Übereinstimmungen sucht, unter Umständen keine für den Endbenutzer befriedigenden Suchergebnisse liefert.

Um diese Probleme anzugehen, soll die hier entwickelte Suchfunktion von der Nutzer-Community generierte Inhalte wie z.B. Beschreibungstexte und Tags als Datenbasis verwenden. Diese Informationen sind, da sie von anderen Nutzern stammen, näher an der Denkwelt des suchenden Nutzers und stellen nach der in Kapitel 2.3 vorgestellten Theorie ab einer ausreichend großen Zahl an Benutzern, die aktiv Inhalte beisteuern eine korrekte und vollständige Beschreibung der Dienste dar. Das hier vorgestellte System zielt auf eine deutlich geringere Benutzerzahl ab als z.B. Wikipedia[64], und dort hat sich gezeigt, dass nur ein kleiner Teil der Benutzer tatsächlich Inhalte erstellt (vgl. Kolbitsch u. Marer[31]). Gerade in der Anfangsphase eines solchen Systems, werden daher voraussichtlich noch nicht genügend von der Benutzergemeinschaft erstellte Beiträge zur Verfügung stehen, um eine korrekte und vollständige Beschreibung sicherzustellen. Die Qualität des Suchsystems wird demnach anfangs relativ gering sein und sich langfristig steigern.

Um auch bei unzureichenden Daten Suchtreffer zu ermöglichen und die bereits besprochenen Unterschiede bei der zur Beschreibung verwendeten Sprache auszugleichen, soll die Suchfunktion in der Lage sein, semantische Ähnlichkeiten zwischen Diensten zu erkennen, und so auch Suchergebnisse zu liefern, in denen die Suchbegriffe zwar nicht wörtlich vorkommen, die aber eine hohe Ähnlichkeit zu einem Ergebnis aufweisen, das die Suchbegriffe enthält.

3.1.3 Relevanz-Feedback und Personalisierung

Suchalgorithmen errechnen die Relevanz eines Suchergebnisses zu einer gegebenen Suchanfrage. Benutzer können nicht immer präzise beschreiben, was genau sie suchen. Entsprechend kann kein Suchalgorithmus garantieren, die Relevanz eines Suchergebnisses genauso einzuschätzen, wie der Benutzer. Die endgültige Entscheidung, ob ein Ergebnis relevant ist oder nicht, liegt beim Nutzer, der dies durch Betrachtung der vorgeschlagenen Suchergebnisse feststellen kann. Derartige Systeme findet man häufig bei Hilfesystemen, bei denen dem Benutzer die Möglichkeit geboten wird, zu einem Suchergebnis zu beurteilen, wie hilfreich die angezeigte Information für ihn ist. Ein weiteres Beispiel ist der Online-Versandhandel Amazon[5]. Dort können Benutzer die Relevanz von Produkt-Rezensionen durch andere Benutzer beurteilen (vgl. Abbildung 3.2). Bei Aufruf einer Produktseite werden die vorhandenen Rezensionen nach ihrer, durch die Benutzergemeinschaft festgelegten Relevanz sortiert.

Abbildung 3.2: Relevanz-Feedback bei Amazon.de[5]

Das zu entwickelnde Web Service Suchsystem soll den Benutzern die Möglichkeit bieten, die Relevanz der Suchergebnisse zu bewerten. Aus diesem Relevanz-Feedback der Benutzer soll das System lernen, um sich dieser Beurteilung langfristig anzunähern. Langfristig wird damit die Entscheidung, ob ein Suchergebnis gut oder schlecht ist, an den Benutzer übergeben.

Nicht nur zwischen Dienstanbietern und -Nutzern gibt es Unterschiede, auch Benutzer untereinander können abhängig von ihrer Aufgabe oder ihrem Umfeld unterschiedliche Begriffe verwenden oder bei der Relevanz-Bewertung einzelner Suchergebnisse voneinander abweichen. Aus diesem Grund soll das Suchsystem nach dem Vorbild von Empfehlungssystemen aus Benutzerprofilen entnommene Attribute verwenden um die Suchergebnisse zu personalisieren. In Kapitel 2.6.1 wurden derartige Systeme vorgestellt, die durch Berücksichtigung von Benutzerinteressen die Qualität von Suchergebnissen steigern können. Dieses Prinzip soll auf das Webservice Suchsystem übernommen werden.

3.1.4 Zusammenfassung

Die folgende Tabelle fasst die Anforderungen an das Suchsystem zusammen:

Funktion	**Anforderungen**
Benutzeroberfläche (vgl. Kapitel 3.1.1)	● Orientierung an Web-Suchmaschinen bei ○ Aussehen ○ Bedienung ● Suchanfragen in natürlicher Sprache ● Mehrzeilige Darstellung der Suchergebnisse: ○ Titelzeile ○ Kurzbeschreibung ○ ggf. weitere Funktionen/Informationen (z.B. Abgabe Relevanz-Feedback, Anzeige Suchscore)
Suchalgorithmus (vgl. Kapitel 3.1.2)	● Community-generierte Inhalte (Beschreibungen, Tags, ...) als Datenbasis ● Beachtung von semantischen Ähnlichkeiten durch Suchalgorithmus
Relevanz-Feedback und Personalisierung (vgl. Kapitel 3.1.3)	● Relevanzbewertung der Suchergebnisse durch Benutzer ● Anpassung der Suchergebnisse an Beurteilung durch Benutzergemeinschaft ● Verwendung von Benutzerprofilen zur Personalisierung der Ergebnisse

Tabelle 3.1: Anforderungen an das Suchsystem

3.2 Entwurf

Auf den folgenden Seiten wird auf Basis der im Grundlagen-Kapitel vorgestellten Ansätze zur Verbesserung des Service-Discovery ein Suchsystem entwickelt, dass die im letzten Abschnitt festgelegten Anforderungen erfüllt. Abbildung 3.3 fasst die Funktionsweise des neuen Suchsystems zusammen.

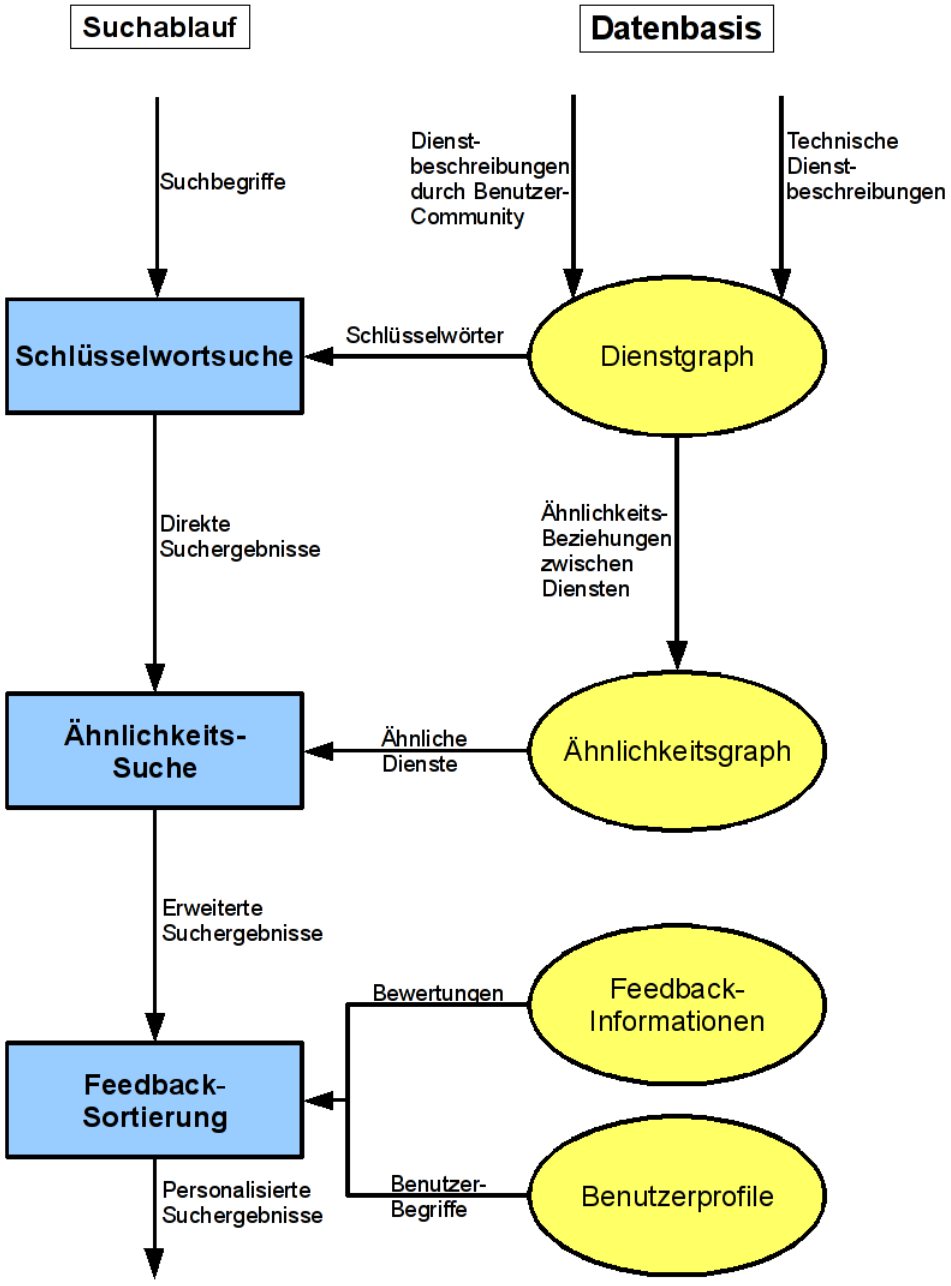

Abbildung 3.3: Aufbau des Suchsystems

Aus Community-generierten Inhalten und technischen Dienstbeschreibungen wird ein Dienstgraph erzeugt. Dieser dient als Datenbasis für eine einfache Schlüsselwortsuche, die Suchanfragen der Benutzer auf syntaktische Übereinstimmungen mit den Beschreibungen durch die Benutzergemeinschaft überprüft. Aus dem Dienstgraphen wird des Weiteren der Ähnlichkeitsgraph berechnet, eine Darstellung von semantischen Ähnlichkeitsbeziehungen zwischen den bekannten Diensten. Dieser wird verwendet, um die direkten Treffer der Schlüsselwortsuche durch weitere Dienste zu ergänzen, die Ähnlichkeiten zu den bereits vorhandenen Ergebnissen aufweisen, ohne allerdings selbst Ergebnisse der Schlüsselwortsuche zu sein. Im letzten Schritt werden durch die Benutzergemeinschaft abgegebene Relevanzbewertungen sowie Benutzerprofile zur Sortierung und Personalisierung der Suchergebnisse verwendet. Die folgenden Abschnitte begründen diesen Aufbau und erklären die in den Komponenten des Suchsystems verwendeten Verfahren und Algorithmen.

3.2.1 Datenbasis

Als Grundlage für die Suchfunktion sollen Community-generierte Dienstbeschreibungen wie z.B. Beschreibungstexte und Schlagworte dienen. Die grundsätzliche Motivation für die Verwendung dieser Daten wurden in Kapitel 2.3 vorgestellt. Benutzer, die bereits Erfahrungen mit der Verwendung eines Dienstes gemacht haben, können diese anderen Benutzern zur Verfügung stellen, um deren Verständnis des Dienstes zu verbessern. Durch andere Benutzer erstellte Informationen eignen sich des Weiteren besser als Datenbasis für eine Suchfunktion, da sie eher die Begriffe verwenden, die von Benutzern in Suchanfragen verwendet werden. Dies kann sich aus einer unterschiedlichen Verwendung von Begriffen ergeben, vor allem aber aus den Perspektiven von Anbietern und Benutzern, die sich stark unterscheiden. Ein Anbieter beschreibt die Funktionalität seines Dienstes, während ein Benutzer nach einer Lösung für einen konkreten Anwendungsfall sucht, den der Anbieter bei der Beschreibung möglicherweise nicht im Sinn hatte.

Nach dem in Kapitel 2.3 im Zusammenhang mit Web 2.0-Technologien vorgestellten Prinzip der selbst-organisierenden Systeme, kann, wenn sich eine ausreichend große Zahl von Benutzern an der Beschreibung von Diensten beteiligt, die Vollständigkeit und Korrektheit der Daten angenommen werden. Im Falle von Dienstbeschreibungen von Anbietern kann insbesondere die Vollständigkeit der Daten nicht vorausgesetzt werden, da Dienste häufig gar nicht, oder nur unzureichend dokumentiert sind, was sowohl das Verständnis durch die Benutzer, als auch die Verwendung als Basis für eine Suchfunktion zusätzlich erschwert.

Zu einem Web Service können verschiedenste Metadaten erfasst werden, die jeden Aspekt

eines Dienstes beschreiben, für eine Suchfunktion eignen sich vor allem solche, die zur Beschreibung der Funktionalität, oder des Umfeldes in dem ein Dienst verwendet wird dienen. Den Namen des Dienstanbieters mit Suchwörtern zu vergleichen kann beispielsweise sinnvoll sein, wenn der Benutzer ganz genau weiß, nach welchem Dienst er Ausschau hält, für eine allgemeine Suche nach einem Dienst, der den Nutzer bei der Erfüllung einer Aufgabe unterstützt, scheinen derartige Informationen nicht relevant zu sein. Für das hier entwickelte Suchsystem wurden auf die vier folgenden Typen von Metadaten ausgewählt, die zur Beschreibung der Funktionalität eines Dienstes und seiner Anwendungsdomäne dienen.

- Beschreibungstexte: Vom Benutzer frei formulierbare, natürlichsprachliche Texte, die für ihn relevante Aspekte des Dienstes beschreiben. Form und Umfang sind nicht klar vorgegeben

- *Tags*: Beliebige Schlagwörter die den Dienst beschreiben. Im Gegensatz zu Beschreibungstexten handelt es sich hier um einzelne Begriffe und keine ausformulierten Texte (vgl. Golder u. Huberman[21])

- *Kategorien*: Diese dienen zur groben Gruppierung von Diensten und können z.B. vom System vorgegeben werden.

- *Kompositionen*: Mehrere Web Services lassen sich mit Hilfe von BPEL zu neuen Diensten orchestrieren, die eine einzelne Schnittstelle zu einer Abfolge von Dienstaufrufen anbieten. Die Betrachtung einer gemeinsamen Verwendung von Diensten innerhalb von Kompositionen kann dazu dienen, Zusammenhänge zwischen verschiedenen Diensten herzustellen.

Im Gegensatz zu einigen kollaborativen Tagging-Systemen im Internet, in denen verschiedene Benutzer dieselben Tags verwenden können und die Häufigkeit der Benutzung durch unterschiedliche Nutzer zur Gewichtung der Begriffe verwendet werden kann (vgl. Golder u. Huberman[21]), verwendet das vorgestellte System eine einfachere Variante eines Tagging-Systems, in dem jedes Schlagwort pro Dienst nur einmal hinzugefügt werden kann. Die Tags werden als Teil der Dienstbeschreibung betrachtet und sind für jeden Benutzer sichtbar, während in Systemen, in denen Tags mehrfach vergeben werden können, jeder Benutzer seine eigenen Tags sieht und mit den Tags anderer Benutzer hauptsächlich über die Suchfunktion in Kontakt kommt. Im World Wide Web sind sowohl Tagging-Systeme mit, als auch ohne Gewichtung (z.B. Seekda.com[53]) verbreitet, hier wurde die einfachere Variante gewählt. Eine Nutzung des TF-IDF-Verfahrens zur Gewichtung der Tags kommt ebenfalls nicht in Frage, da jedes Schlüsselwort eines Dienstes nur genau einmal vorkommt, damit also die Termfrequenz jedes Wortes gleich ist. Der zweite Teil von TF-IDF, die inverse

Dokumenthäufigkeit dient dazu, Wörter die häufig, in vielen Dokumenten vorkommen niedriger zu bewerten. Die Aussagekraft dieses Wertes ist im vorliegenden Fall jedoch ebenfalls fraglich, da zum einen jedes Schlagwort höchstens einmal pro Beschreibung vorkommt und zum anderen angenommen werden kann, dass bei Tags im Gegensatz zu natürlichsprachlichen Texten hauptsächlich für die Beschreibung relevante Begriffe verwendet werden.

Beschreibungstexte sind natürlichsprachliche Texte, auf die das TF-IDF-Verfahren zur Gewichtung der enthaltenen Begriffe angewendet werden kann. Ein Problem stellt allerdings der möglicherweise geringe Umfang dieser Texte dar. TF-IDF kann bei der Analyse von langen Textdokumenten zuverlässig die Wichtigkeit einzelner Wörter berechnen, Dienstbeschreibungen von Benutzern bestehen aber möglicherweise nur aus einem kurzen Satz, in dem in der Regel kaum Begriffe mehrfach verwendet werden.Das Problem von unterschiedlich langen Texten kann in TF-IDF zwar mit Hilfe des Normalisierungsfaktors (vgl. Salton u. Buckley[52]) vermindert werden, wenn eine Beschreibung allerdings nur aus einem Satz besteht, entsteht eine mit den Tags vergleichbare Situation.

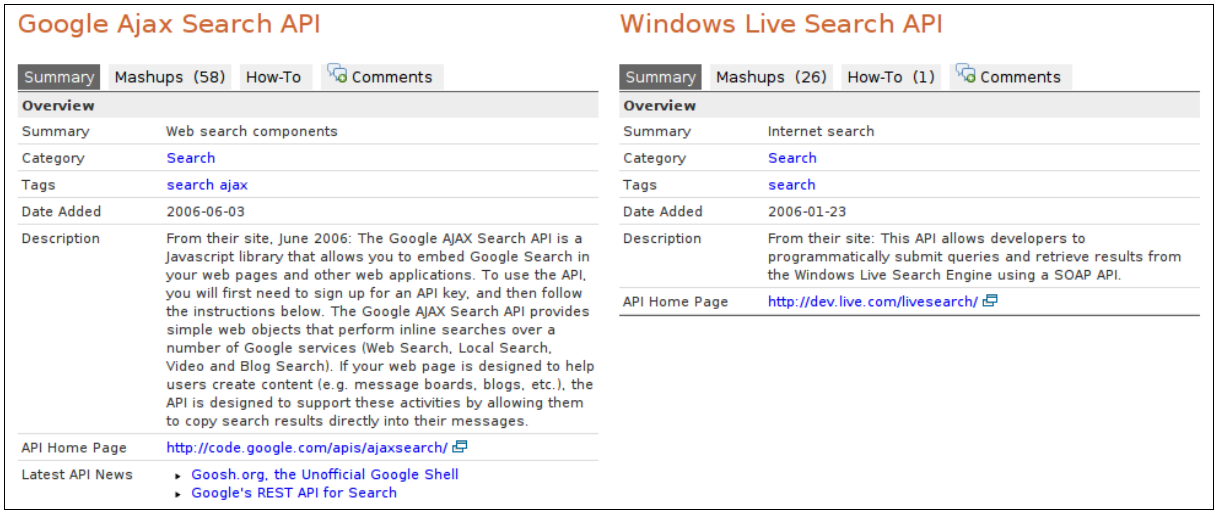

Abbildung 3.4: Beispiele für unterschiedlichen Umfang von Dienstbeschreibungen

Abbildung 3.4 zeigt Beispiele für zwei unterschiedlich lange Dienstbeschreibungen auf ProgrammableWeb.com[49], die zeigen, wie schwer Form und Umfang von Dienstbeschreibungen und somit auch die Anwendbarkeit von Verfahren wie TF-IDF vorhersagen lässt. Da eine Mindestlänge von Beschreibungstexten nicht vorausgesetzt werden kann, kann auch nicht sicher angenommen werden, dass sich das TF-IDF-Verfahren zu einer sinnvollen Gewichtung der Suchbegriffe eignet. Aus diesem Grund werden die Begriffe aus den Beschreibungstexten nach der Entfernung von Stoppwörtern auf die gleiche Weise weiterverwendet wie Tags. Aus Tags und Beschreibungstexten wird somit für jeden Dienst eine gemeinsame, ungewichtete Schlüsselwortmenge erzeugt, in der jeder Begriff

höchstens einmal vorkommt.

Kategorien dienen ebenfalls der Beschreibung von Diensten und werden auf die gleiche Weise verwendet, wie Tags und aus Beschreibungstexten extrahierte Schlüsselwörter. Allerdings gibt nicht wie bei Tags und aus Beschreibungstexten extrahierten Begriffen mehr Schlüsselwörter als Dienste, sondern es gibt eine kleine Zahl von Kategorien, denen sämtliche Dienste zugeordnet werden können. Kategorien dienen also einer groben Klassifizierung von Diensten. Die Kategorien können dabei vom System vorgegeben werden. Die Zuweisung der Dienste zu Kategorien geschieht durch die Benutzer, genau wie dies bei den anderen verwendeten Metadaten der Fall ist. Eine Anforderung an das Suchsystem ist die Einbeziehung von semantischen Ähnlichkeiten. Dies wird durch Kategorien unterstützt, da sie zur Gruppierung von Objekten dienen, und sich so auch dann eine schwache Verwandtschaft zwischen Diensten herleiten lässt, die zu derselben Kategorie gehören, wenn die sonstige Dienstbeschreibung nicht übereinstimmt. Die programminterne Verwendung der Kategorien gleicht der von anderen Schlüsselwörtern, der Unterschied besteht darin, dass eine geringe Anzahl von Kategorien eine möglicherweise große Anzahl von Diensten beschreibt, während dieses Verhältnis bei den anderen verwendeten Schlüsselwörtern umgekehrt ist, da anzunehmen ist, dass die Beschreibung eines Dienstes i.d.R aus mehr als einem Wort besteht.

Auf BPEL basierende Kompositionen können genau wie Dienste mit Beschreibungstexten, Tags und Kategorien versehen werden, da sie nach außen hin eine eigene Webservice-Schnittstelle anbieten, also als eigenständiger Dienst betrachtet und genutzt werden können. Sie eignen sich jedoch auch um ähnlich wie Kategorien Dienste oder Operationen zu gruppieren. Die gemeinsame Nutzung sagt aus, dass die verwendeten Dienste oder Operationen Teil desselben Prozesses sein können. Wenn sich ein Benutzer für einen Dienst interessiert kann daraus geschlossen werden, dass ein Dienst der in der Vergangenheit mit diesem zusammengearbeitet hat ebenfalls von Interesse sein kann.

Kompositionen bestehen aus einer Abfolge von Operationsaufrufen. Die verwendeten Operationen können dabei aus verschiedenen Diensten stammen. Ein Dienst enthält häufig mehrere verwandte Operationen, die nicht unbedingt alle innerhalb einer Komposition verwendet werden. Für den Benutzer, der eine Komposition erstellen möchte, sind die Operationen daher von größerem Interesse, als die Dienste selbst. Aus diesem Grund wird das Suchsystem, im Gegensatz zu den meisten webbasierten Webservice Suchsystemen wie z.B. Seekda.com[53], bevorzugt auch Operationen als Suchergebnisse liefern, die entsprechend auch separat mit Beschreibungstexten versehen werden können.

Aus der Verwendung von Kompositionen, atomaren Diensten und den Operationen einzelner

Dienste als mögliche Suchergebnisse ergibt sich eine Hierarchie, deren Elemente unabhängig von ihrer Ebene in der Hierarchie durch Schlüsselwörter beschrieben werden können, die aus einer Kombination von Tags, Kategorien und aus Beschreibungstexten extrahierten Begriffen bestehen. In Abbildung 3.5 sieht man eine grafische Darstellung der Hierarchie der vorhandenen Daten.

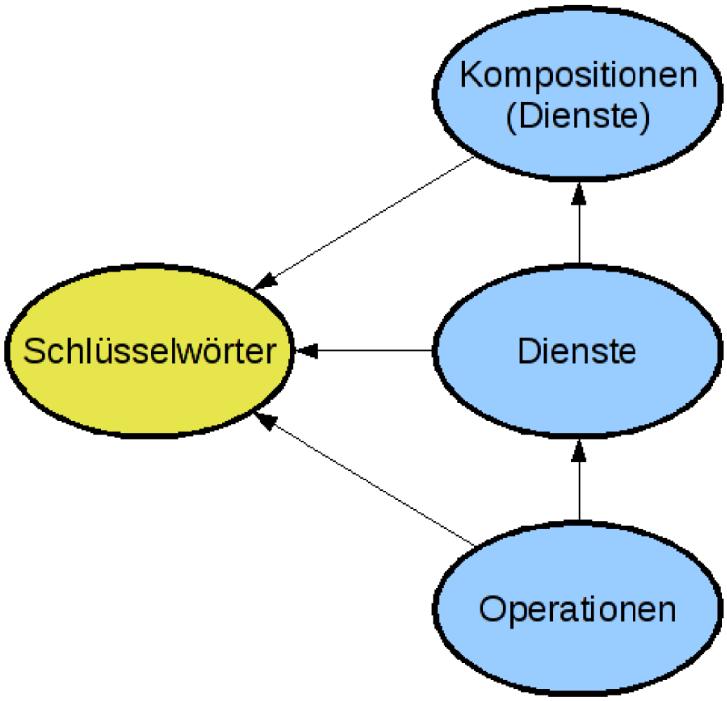

Abbildung 3.5: Hierarchie der verwendeten Daten

Dabei ist anzumerken, dass die Zusammenfassung von Diensten zu Kompositionen eine Vereinfachung der Realität entspricht, da wie bereits festgestellt wurde BPEL-Kompositionen aus einer Abfolge von Operationsaufrufen bestehen, und nicht einer Zusammenfassung von vollständigen Diensten. Eine direkte Zuordnung von verwendeten Operationen zu Kompositionen ist schwierig, da zwischen verwendeten und von der Komposition angebotenen Operationen unterschieden werden müsste. Es bleibt daher eine offene Frage, wie sich eine Beziehung zwischen Operationen, die innerhalb von Kompositionen zusammenarbeiten, am sinnvollsten darstellen lässt. Die hier gewählte, vereinfachte Darstellung dieses Zusammenhangs eignet sich jedoch, um Kompositionen innerhalb der in den folgenden Abschnitten vorgestellten Suchalgorithmen zu berücksichtigen.

3.2.2 Such- und Ranking-Algorithmen

In Abbildung 35 wurde bereits die Aufteilung des Suchsystems in eine Schlüsselwort- und eine Ähnlichkeitssuche vorgestellt. In diesem Abschnitt soll dieses Vorgehen begründet und

die Funktionsweise der beiden Module, sowie der dazugehörigen Datenstrukturen erklärt werden.

Die wichtigste Anforderung an die Suchfunktion ist die Fähigkeit, auch Ergebnisse zu berechnen, in denen die gesuchten Begriffe nicht wörtlich vorkommen. Um dies zu erreichen, muss ein Verfahren zur Berechnung der Ähnlichkeit zwischen Diensten realisiert werden. Die Verwendung von Semantic Web Services erscheint aus den in Kapitel 2.2.3 dargestellten Gründen als nicht praktikabel, insbesondere weil Aufbau und Wartung von Ontologien aufwändig ist und das eigentliche Anwendungsfeld dieser Technologie nicht die Auswertung von einfachen, benutzergenerierten Suchbegriffen ist. Des Weiteren würde die Benutzung eines solchen Systems durch die Notwendigkeit der expliziten semantischen Beschreibung durch den Anbieter oder Benutzer weiter erschwert. Die tatsächlichen Vorteile von SWS würden nicht zum Tragen kommen und das Erweitern von Suchergebnissen auf semantisch ähnliche Objekte lässt sich auch auf einfacherem Wege, z.B. mit LSI realisieren. Die Verwendung von LSI setzt allerdings einen Mechanismus zur Gewichtung von Schlüsselwörtern voraus. Derartige Verfahren wie z.B. TF-IDF lassen sich, wie im vorherigen Abschnitt festgestellt, im Zusammenhang mit den vorliegenden, möglicherweise uneinheitlichen und unvollständigen Informationen nicht rechtfertigen, was die Verwendung von LSI behindern und schwer begründbar machen würde. Des Weiteren sollte die im vorherigen Abschnitt beschriebene Hierarchie der Datenbasis berücksichtigt werden, um zwischen Diensten und Operationen unterscheiden zu können. Spätere Benutzer verwenden einzelne Operationen eines Dienstes und nicht den gesamten Dienst, entsprechend sollte zwischen Diensten und Operationen unterschieden werden und Operationen als Suchergebnisse in Frage kommen. Diese Voraussetzung ist bei LSI ohne eine explizite Erweiterung um diese Funktion nicht erfüllt, da sich das Verfahren ausdrücklich von Hierarchie-basierten Suchverfahren abgrenzt (vgl. Deerwester et al.[15]).

Das in Kapitel 2.5.2 vorgestellte Ähnlichkeitsgraph-Verfahren ist ursprünglich zur Ähnlichkeitsberechnung innerhalb von Teilgraphen einer Ontologie bestimmt (vgl. Andreasen et al.[7]), lässt sich jedoch allgemein auf hierarchisch angeordnete Informationen übertragen, in denen Generalisierungs- bzw. Spezialisierungsbeziehungen vorkommen. Das Verfahren berechnet die Ähnlichkeit grundsätzlich als relative Übereinstimmung der erreichbaren Knoten, also im vorliegenden Fall der vorhandenen Schlüsselwörter. Durch die Darstellung der Daten in einem gerichteten Graphen, kann dieses ansonsten einfache Verfahren die Hierarchie der betrachteten Daten berücksichtigen. Dienste stellen Bündel von Operationen dar. Eine Dienstbeschreibung beschreibt die gemeinsame Domäne aller enthaltenen Operationen, während sich Operationsbeschreibungen auf die Funktionalität einzelner Operationen beziehen. Eine Dienstbeschreibung kann somit als Generalisierung einer Operationsbeschreibung verstanden werden. Betrachtet man z.B. einen Dienst, der

Informationen über das Wetter liefert, so sollten die Operationsbeschreibungen für jede Operation Informationen darüber enthalten, welche Daten genau geliefert werden und welche Eingabedaten erforderlich sind. Die Dienstbeschreibung müsste hingegen wesentlich allgemeiner formuliert sein und aussagen, dass der Dienst verschiedene Operationen zur Abfrage von Wetterinformationen bietet, ohne im Detail auf die einzelnen enthaltenen Operationen einzugehen. Ähnlich wie im Beispiel aus Abbildung 2.4 ‚in dem „Tier“ eine Generalisierung von „Hund“ darstellt, wäre „Informationen über das Wetter“ in einer Dienstbeschreibung allgemeiner als z.B. „Wetterbericht für die nächsten 5 Tage anhand einer Postleitzahl“ in einer Operationsbeschreibung. Die Verwendung der in Kapitel 2.5.2 vorgestellten Formel zur Berechnung der Ähnlichkeit in einem gerichteten Graphen führt dazu, dass die Beschreibung von generelleren Knoten im Graphen in die Berechnung der Ähnlichkeit von spezielleren Knoten einbezogen werden. Da der Graph gerichtet ist gilt dies nicht für die Gegenrichtung. Auf Webservices bezogen bedeutet dies, dass die Dienstbeschreibung, die eine Beschreibung der gemeinsamen Domäne aller im Dienst enthaltenen Operationen darstellt, einen Einfluss auf die Ähnlichkeit einer Operation zu anderen Operationen oder Diensten hat. Im Gegensatz dazu wirken sich Operationsbeschreibungen nicht auf den übergeordneten Dienst aus. Diese Eigenschaft ist deshalb wichtig, weil die Operationsbeschreibungen im Gegensatz zur Dienstbeschreibung auch wirklich nur für die jeweils beschriebenen Operationen relevant sind und sich nicht auf alle anderen Operationen des Dienstes übertragen lässt.

Andreasen et al.[7] begründen die von ihnen verwendete Formel mit drei Eigenschaften, die für hierarchische Informationen innerhalb einer Ontologie gelten sollten. Diese Eigenschaften lassen sich auf den Anwendungsfall übertragen. Abbildung 3.6 zeigt ein Beispiel für einen Dienstgraphen, der über zwei Dienste (D1 und D2) verfügt, die jeweils über eine Operation (O1 und O2) verfügen und mit insgesamt zwei Schlüsselwörtern (SW1 und SW2) beschrieben werden.

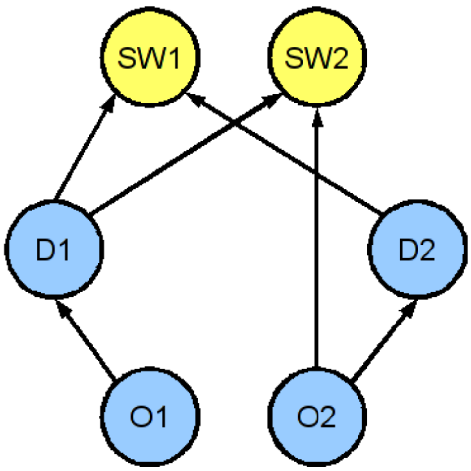

Abbildung 3.6: Beispiel Dienstgraph

Die drei in Kapitel 2.5.2 vorgestellten Eigenschaften lassen sich wie folgt auf den Dienstgraphen übertragen:

- *Generalisierungskosten: Eine Generalisierung sollte sich negativer auf die Ähnlichkeit auswirken als eine Spezialisierung.* Für Webservices bedeutet dies, dass die Ähnlichkeit von einer Operation zu einem Dienst höher eingeschätzt wird, als die entgegengesetzte Ähnlichkeit vom Dienst zur Operation. Allgemein sollten Operationen also vor Diensten bevorzugt werden. Dies bedeutet jedoch nicht, das grundsätzlich gelten muss, dass die Ähnlichkeit einer Operation zu einem Dienst größter sein muss als die umgekehrte Richtung, nur dass ein Dienst weniger Übereinstimmungen braucht, um dieselbe Ähnlichkeit zu erreichen. Diese leichte Bevorzugung von Operationen wird dadurch erreicht, dass von ihnen aus durch die niedrigere Position in der Hierarchie mehr Knoten erreicht werden können. In Abbildung 3.6 ist z.B. vom Knoten O2 aus direkt nur SW2 erreichbar, zusätzlich jedoch auch noch der zur Operation gehörende Dienst D2 und über diesen auch SW1. Insgesamt sind von O2 aus inklusive dem Knoten O2 selbst vier Knoten erreichbar. Der Dienst D1 wird mit beiden Schlüsselwörtern beschrieben, insgesamt sind somit drei Knoten erreichbar. Die beiden Schlüsselwörter SW1 und SW2 sind von D1 und O2 aus erreichbar. Verwendet man zur Berechnung der Ähnlichkeit die Formel $\mathrm{sim}(x,y)=\frac{|\alpha(x)\cap\alpha(y)|}{|\alpha(x)|}$ (vgl. Andreasen et al.[7]), die wie in Kapitel 2.5.2 festgestellt wurde die Generalisierungskosteneigenschaft bei Ähnlichkeitsgraphen erfüllt, so erhält man:

  sim(D1, O2) = 2/3

  sim(O2, D1) = 2/4 = 1/2

  Um zu erreichen, dass sim(O2, D1) größer ist als sim(D1, O2) müsste der Dienst D1 um mindestens drei weitere Schlüsselwörter verfügen, die von O2 aus nicht erreichbar sind. Eine Operation braucht also weniger eigene Schlüsselwörter um eine höhere Ähnlichkeit zu erreichen, als ein Dienst, die Generalisierungskosten sind somit realisiert.

- *Spezifitätskosten: Ähnlichkeiten zwischen generellen Knoten sollten geringer gewertet werden als zwischen spezielleren Knoten*. Diese Eigenschaft wird ebenfalls durch die Hierarchie unterstützt. Operationen erben die Beschreibungen der Dienste, daher brauchen sie weniger direkte Übereinstimmungen um als ähnlich befunden zu werden. Vergleicht man zwei Operationen, die sowohl über eigene, als auch über

geerbte Schlüsselwörter verfügen ist die Chance größer eine Übereinstimmung zu finden, als wenn wie bei Diensten nur die eigenen Schlüsselwörter verwendet werden. Diese Eigenschaft wird jedoch durch die geringe Anzahl an Hierarchieebenen im Dienstgraphen und die Tatsache, dass sowohl Dienste als auch Operationen auf dieselben Schlüsselwörter verweisen können und somit die Hierarchie teilweise aufheben abgeschwächt. Entsprechend werden Spezifitätskosten in der Realität kaum in Erscheinung treten und sie sind anhand eines einfachen Beispiels schwer zu demonstrieren. Dies führt dazu, dass die Ähnlichkeit zwischen zwei Operationen nicht grundsätzlich höher gewertet wird als die Ähnlichkeit zwischen zwei Diensten, was durchaus positiv ausgelegt werden kann, da damit dem Inhalt der jeweiligen Beschreibungen mehr Bedeutung eingeräumt wird als der Position in der Hierarchie.

- Spezialisierungskosten: *Eine weitere Spezialisierung senkt die Ähnlichkeit.* Diese Eigenschaft ist insbesondere innerhalb von Diensten von Bedeutung. Berechnet man z.B. anhand der oben verwendeten Formel im Beispiel sim(D2, D1) und sim(D2, O1), so stellt man fest, dass sich in beiden Fällen eine Ähnlichkeit von 0,5 ergibt. Das bedeutet, dass es, wenn eine Ähnlichkeit zu einem Dienst berechnet wird und es keine zusätzliche, direkte Übereinstimmung mit einer seiner Operationen gibt, die Ähnlichkeit zu einem Dienst und zu allen seinen Operationen identisch ist. Zwar sollen Operationen grundsätzlich gegenüber Diensten bevorzugt werden, wenn allerdings eine Suchanfrage so allgemein formuliert ist, dass sie mit einer Dienstbeschreibung übereinstimmt, aber mit keiner spezielleren Operationsbeschreibung, dann sollte der Dienst ein besseres Ergebnis darstellen, als die dazugehörigen Operationen. Andreasen et al.[7] führen zur Berücksichtigung der Spezialisierungskosten die folgende Formel ein:

$$\mathrm{sim}(x,y) = p\frac{|\alpha(x)\cap\alpha(y)|}{|\alpha(x)|} + (1-p)\frac{|\alpha(x)\cap\alpha(y)|}{|\alpha(y)|}$$

Diese Formel wertet im vorliegenden Anwendungsfall die Ähnlichkeit von Operationen im Vergleich zu Diensten ab. Der Faktor p bestimmt, wie stark sich die Spezialisierungskosten auf das Ergebnis auswirken(vgl. Kapitel 2.5.2). Unter Verwendung dieser Formel mit p=4/5 ergibt sich:

$$\mathrm{sim}(D2, D1) = \frac{4}{5}*\frac{1}{2} + \frac{1}{5}*\frac{1}{3} = \frac{7}{15} \approx 0{,}47$$

$$\mathrm{sim}(D2, D1) = \frac{4}{5}*\frac{1}{2} + \frac{1}{5}*\frac{1}{4} = \frac{9}{20} = 0{,}45$$

In diesem Ergebnis sind die Spezifitätskosten berücksichtigt.

Die von Andreasen et al.[7] verwendeten Formel stellt demnach eine geeignete Methode zur Berechnung der Ähnlichkeit von Diensten und Operation im Dienstgraphen dar. Allerdings gibt es einige entscheidende Unterschiede zwischen dem vorliegenden Dienstgraphen und den dem Verfahren zugrunde liegenden Ähnlichkeitsgraphen, die eine Verwendung als Basis für eine Suchfunktion einschränken. Die ursprünglichen Ähnlichkeitsgraphen verfügen über eine vollständige Hierarchie, mit einem „Wurzelknoten“, der von allen anderen Knoten aus erreichbar ist. Im Falle des Dienstgraphen existiert eine derartige Struktur nicht, und um diese künstlich zu erzeugen, müsste der Dienstgraph um zusätzliche Knoten erweitert werden, die im Gegensatz zu den Knoten in einer Ontologie keinen realen Objekten oder Konzepten entsprechen. Diese Knoten ohne eigene Bedeutung würden den Einfluss der tatsächlichen Beschreibung auf die berechnete Ähnlichkeit verringern. So wären sich z.B. Dienste alleine aufgrund der Tatsache, dass sie Dienste sind, ähnlicher als vorher, ohne dass es mehr Übereinstimmungen bei der Beschreibung gibt. Je mehr zusätzliche Ebenen eingeführt werden, desto geringer wirken sich die Dienstbeschreibungen auf die Ähnlichkeitsberechnung aus, weshalb grundsätzlich auf eine derartige übergeordnete Struktur verzichtet wurde. Dies führt jedoch dazu, dass sich mit Hilfe der in Kapitel 2.5.2 vorgestellten Formel nicht die Ähnlichkeit von jedem Knoten zu jedem anderen, sondern nur die direkter Nachbarn mit mindestens einem übereinstimmenden Schlüsselwort berechnen lässt. Eine weitere Schwierigkeit liegt darin, dass im Ähnlichkeitsgraphen alle Knoten für Konzepte stehen, zwischen denen eine Ähnlichkeit berechnet werden kann. Im Dienstgraphen gibt es zwei verschiedene Arten von Knoten, Objekte, die als Suchergebnisse in Frage kommen und Schlüsselwörter, die diese beschreiben. Eine Berechnung der Ähnlichkeit zwischen Schlüsselwörtern ist für die Ermittlung der Suchergebnisse nicht von Belang, da auf Objekte als Suchergebnisse abgezielt wird. Außerdem ist eine derartige Berechnung grundsätzlich ausgeschlossen, da Schlüsselwörter keine ausgehenden Kanten und dementsprechend auch keine gemeinsam erreichbaren Knoten haben. Diese Eigenschaften von Dienstgraphen schließen die Verwendung des von Andreasen et al. [7] vorgeschlagenen Verfahrens zur Auswertung von Suchanfragen mit Ähnlichkeitsgraphen aus. In diesem Verfahren sind die vom Benutzer verwendeten Suchbegriffe und die gesuchten Schlüsselwörter jeweils Konzepte innerhalb einer Ontologie, zwischen denen direkt eine Ähnlichkeit berechnet werden kann. Im Dienstgraphen sind die Schlüsselwörter, nach denen gesucht wird nicht die Objekte, die letztendlich Ergebnisse darstellen.

Die Tatsache, dass nur zwischen direkt benachbarte Knoten eine Ähnlichkeit berechnet werden kann, stellt für das Suchsystem kein großes Problem dar, da Dienste, die im Graphen weit voneinander entfernt liegen, deren Beschreibungen sich also nicht überschneiden, voraussichtlich eine so geringe Ähnlichkeit hätten, dass sie ohnehin keine

relevanten Suchergebnisse darstellen würden. Diese Eigenschaft führt jedoch dazu, dass das in Abbildung 3.3 gezeigte zweistufige Suchverfahren angewendet werden muss, das Ergebnisse einer einfachen Schlüsselwortsuche um über das Ähnlichkeitsgraph-Verfahren berechnete, ähnliche Ergebnisse erweitert. Schlüsselwortbasierte Suchverfahren bauen normalerweise auf gewichteten Schlüsselwortmengen auf, um Ergebnisse zu erzielen, die der menschlichen Einschätzung möglichst ähnlich sind (vgl. Salton u. Buckley[52]). Da, wie bereits festgestellt wurde, im vorliegenden Fall keine gewichteten Begriffe verwendet werden, greift das Suchsystem für den ersten Schritt auf einen einfachen Suchalgorithmus zurück, der anhand der nachfolgenden Formel die Relevanz der Dienst- bzw. Operationsbeschreibungen für eine Suchanfrage errechnet:

$$\text{Punktzahl} = \frac{\text{Anzahl in Beschreibung vorkommender Suchwörter}}{\text{Gesamtzahl Suchwörter}}$$

Die Formel berechnet den Anteil der gesuchten Begriffe, der in einer Dienst- oder Operationsbeschreibung vorkommt. Wenn z.B. die Suchanfrage aus vier Begriffen besteht und zwei davon Teil einer Beschreibung sind, ergibt sich eine Punktzahl von „0,5“. Die Punktzahl liegt also zwischen „1“, falls alle Suchbegriffe Teil einer Dienstbeschreibung sind, und „0“ falls keine Übereinstimmung vorliegt. In diesem ersten Schritt der Suche wird die Hierarchie der Daten noch nicht berücksichtigt. Operationen werden nur dann als Suchergebnisse geliefert, wenn die eigene Operationsbeschreibung Übereinstimmungen mit den Suchbegriffen aufweist.

Objekt	**Ist Teil von**	**Beschreibung**
Dienst1	-	Schlüsselwort1 Schlüsselwort2 Schlüsselwort3
Dienst2	Dienst3	Schlüsselwort2
Dienst3	-	Schlüsselwort3 Schlüsselwort4
Dienst4	-	Schlüsselwort5
Operation1	Dienst1	Schlüsselwort1
Operation2	Dienst1	-
Operation3	Dienst2	Schlüsselwort5
Operation4	Dienst3	Schlüsselwort4
Operation5	Dienst4	Schlüsselwort4

Tabelle 3.2: Beispiel: Ausgangssituation

Tabelle 3.2 beschreibt ein Beispiel, dessen Graph in Abbildung 3.7 dargestellt wird. Das Beispiel enthält vier Dienste mit den Namen „Dienst1“ bis „Dienst4“ (im Folgenden auch abgekürzt als „D1“ bis „D4“) mit den fünf Operationen „Operation1“ bis „Operation5“ (bzw. „O1“ bis „O5“), wobei jeder Dienst über ein bis zwei Operationen verfügt. Des Weiteren ist

„Dienst2“ Teil der Komposition „Dienst3“. Da auf BPEL basierende Kompositionen, wie bereits festgestellt wurde, nach außen hin als Dienste auftreten und eigene Operationen anbieten, wird im Graphen nicht explizit zwischen Diensten und Kompositionen unterschieden. Die Dienste und Operationen werden jeweils durch einen Teil der insgesamt fünf verwendeten Schlüsselwörter beschrieben („Schlüsselwort1“ bis „Schlüsselwort5“).

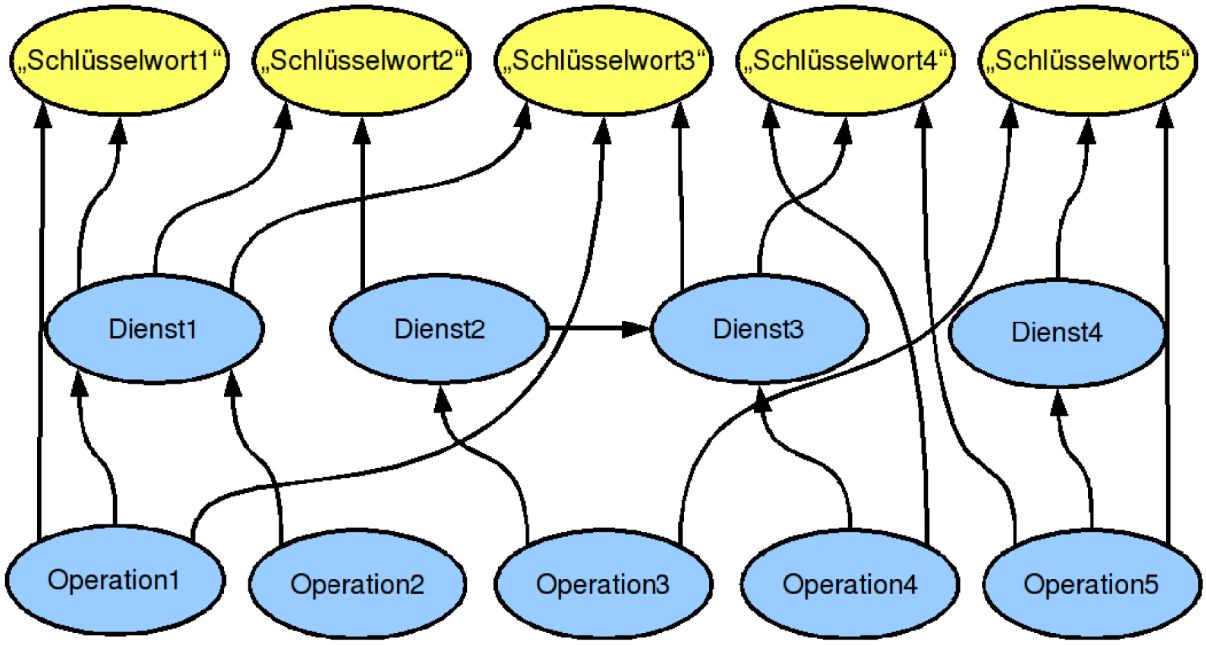

Abbildung 3.7: Beispiel: Dienstgraph

Wenn die Suchanfrage „Schlüsselwort1 Schlüsselwort2“ in diesem Beispiel durchgeführt wird, ergibt sich für D1 eine Punktzahl von 1,0, da beide Suchbegriffe Teil der Beschreibung sind. Für D2 und O1 ergibt sich jeweils ein Treffer von 0,5, da nur einer der beiden Begriffe vorkommt. Alle anderen Knoten haben keine Verbindung zu einem der Suchwörter und erhalten daher eine Punktzahl von 0. Dies gilt auch für die Operationen von D1 und D2, deren Operationsbeschreibungen keine Übereinstimmungen mit den Suchbegriffen aufweisen, da während der Schlüsselwortsuche noch keine „Vererbung“ von Dienstbeschreibungen an Operationen geschieht.

Ein Aspekt von Suchmaschinen, der bisher nicht betrachtet wurde, ist die Frage, ob teilweise Übereinstimmungen von Suchbegriffen berücksichtigt werden sollen. Zu diesem Zweck könnten beispielsweise Wildcards, also Platzhalter für beliebige andere Zeichen in der Suchanfrage eingesetzt werden. Da teilweise Übereinstimmungen und Wildcards zu irrelevanten Suchergebnissen führen könnten und des Weiteren bei Web-Suchmaschinen, die das Vorbild für dieses System darstellen, Wildcards nicht verbreitet sind, wird stattdessen das in Kapitel 2.4.1 vorgestellte Stemming-Verfahren verwendet (vgl. Caumanns [14]) um einen ähnlichen Effekt zu erreichen. Alle Schlüsselwörter im Dienstgraphen sind die

gekürzten Grundformen der Begriffe aus den Beschreibungstexten bzw. der Tags. Die vom Benutzer eingegebenen Suchbegriffe werden ebenfalls auf ihre Grundform reduziert, was dazu führt, dass unabhängig von der genutzten Form eines Wortes in Beschreibung und Suchanfrage Treffer geliefert werden.

Im zweiten Schritt der Suche wird mit Hilfe der von den Ähnlichkeitsgraphen und in diesem Kapitel bereits auf den Dienstgraphen übertragenen Formel

$$\mathrm{sim}(x,y) = p\frac{|\alpha(x)\cap\alpha(y)|}{|\alpha(x)|} + (1-p)\frac{|\alpha(x)\cap\alpha(y)|}{|\alpha(y)|}$$ (vgl. Andreasen et al.[7])

die Ähnlichkeit zwischen allen Diensten bzw. Operationen berechnet, die mindestens ein gemeinsames Schlüsselwort enthalten. Für den Faktor p wird im Folgenden der von Andreasen et al.[7] vorgeschlagene Wert von *4/5* verwendet. Wird eine Änderung am Dienstgraphen durchgeführt, also z.B. ein neuer Dienst hinzugefügt oder eine Beschreibung verändert, werden mit Hilfe dieser Formel die Ähnlichkeiten zwischen allen Knoten neu berechnet. Tabelle 3.3 zeigt das Ergebnis dieser Berechnung für das oben genannte Beispiel in Form einer Matrix, die Ähnlichkeiten jedes Knotens zu jedem anderen enthält.

	D1	**D2**	**D3**	**D4**	**O1**	**O2**	**O3**	**O4**	**O5**
D1	**1**	0,5	0,27	0	0,96	0,96	0,46	0,25	0
D2	0,5	**1**	0,53	0	0,48	0,48	0,91	0,5	0,25
D3	0,32	0,63	**1**	0	0,31	0,31	0,89	0,95	0,32
D4	0	0	0	**1**	0	0	0,43	0	0,9
O1	0,84	0,42	0,23	0	**1**	0,8	0,38	0,21	0
O2	0,84	0,42	0,23	0	0,8	**1**	0,38	0,21	0
O3	0,33	0,66	0,54	0,21	0,31	0,31	**1**	0,49	0,33
O4	0,25	0,5	0,8	0	0,24	0,24	0,69	**1**	0,25
O5	0	0,25	0,27	0,6	0	0	0,46	0,25	**1**

Tabelle 3.3: Beispiel: Ähnlichkeitsmatrix

Eine Ähnlichkeit von „0“ bedeutet, dass aufgrund der zuvor beschriebenen Unterschiede zum ursprünglichen Ähnlichkeitsgraph-Verfahren keine Ähnlichkeit berechnet werden konnte, da die verglichenen Knoten über keine gemeinsam in Kantenrichtung erreichbaren Knoten verfügen. Stellt man diese Matrix als Graph dar, so erhält man einen neuen Dienstgraphen, der keine Schlüsselwörter mehr enthält. Stattdessen sind die Kanten zwischen den Diensten bzw. Operationen gewichtet, wobei das Gewicht der Ähnlichkeit zwischen den verbundenen Knoten entspricht. Die Ähnlichkeit ist dabei, wie zuvor festgestellt wurde, nicht symmetrisch. So sieht man im Beispiel, dass sim(D1, D3)=0,27 und sim(D3, D1)=0,32 ist. Dies ergibt sich

daraus, dass von D1 aus insgesamt vier Knoten erreichbar sind, von D3 aus nur drei. Die Übereinstimmung zwischen D1 und D3 ist daher aus Sicht von D3 von größer. Abbildung 3.8 zeigt den errechneten Graphen. Aus Gründen der Übersichtlichkeit wurde auf die Darstellung der Kantengewichte verzichtet, die Abbildung zeigt also lediglich zwischen welchen Knoten eine Ähnlichkeit festgestellt werden kann. Im Beispiel gibt es mehr Dienste und Operationen als Schlüsselwörter, in der Realität ist dieses Verhältnis in der Regel umgekehrt, da Dienst- bzw. Operationsbeschreibungen für gewöhnlich aus mehreren Wörtern bestehen, und nicht alle Dienste ähnliche Aufgaben erfüllen, was zu größeren Unterschieden in den Dienstbeschreibungen führt. Dies führt dazu, dass zwischen dem Großteil der Knoten keine Kanten existieren, diese Knoten also untereinander eine Ähnlichkeit von „0“ aufweisen und nicht zur Berechnung von Suchergebnissen verwendet werden können. Im Beispiel wurde die Anzahl der Schlüsselwörter reduziert um eine gewisse Übersichtlichkeit zu wahren und weil die verwendete Schlüsselwortzahl zur Demonstration aller relevanten Konzepte ausreicht.

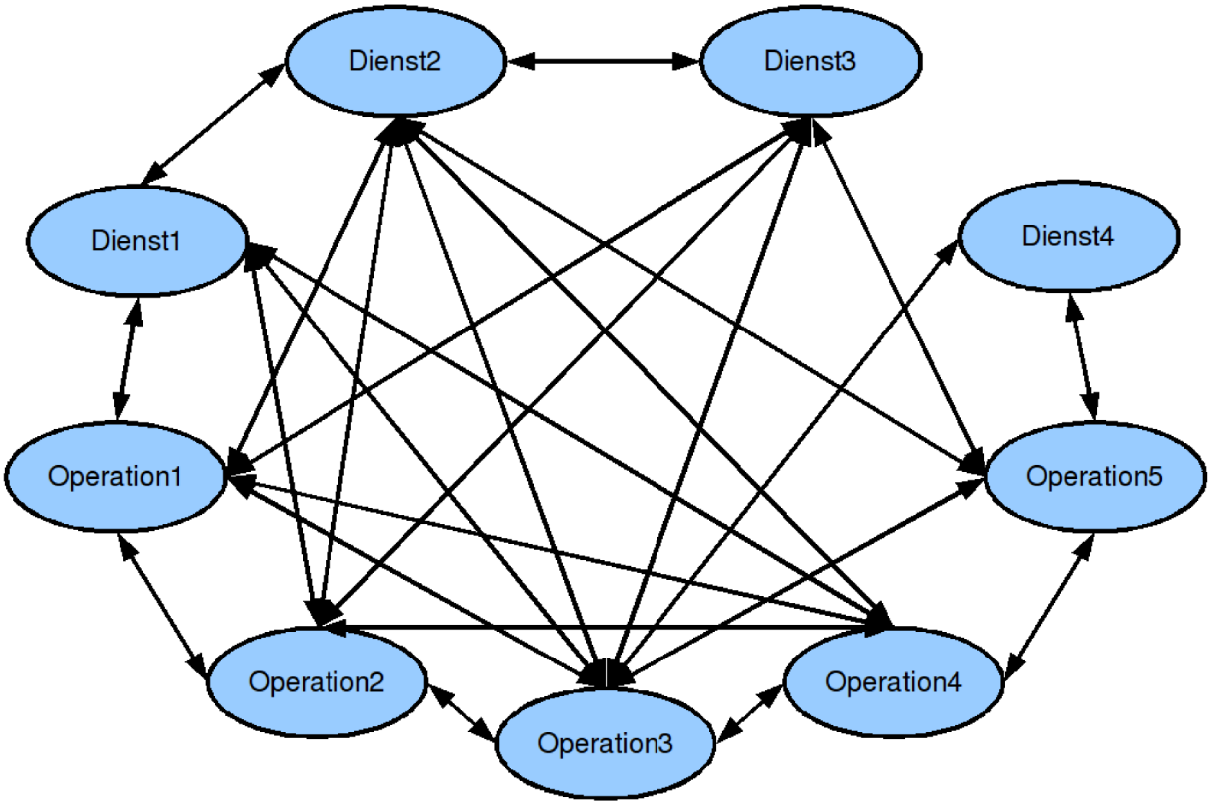

Abbildung 3.8: Beispiel: Ähnlichkeitsbeziehungen zwischen Knoten

Um die Schlüsselwort- und Ähnlichkeitssuche zu kombinieren, werden nun zu jedem Ergebnis der Schlüsselwortsuche ähnliche Dienste gesucht. Die Ähnlichkeit zu einem anderen Suchergebnis reicht alleine nicht aus, um eine Aussage über die Relevanz für eine Suchanfrage zu treffen, da sie nur die Ähnlichkeit zu einem bereits bekannten Suchergebnis darstellt. Die Ergebnisse von Schlüsselwort- und Ähnlichkeitssuche müssen demnach zu

einem Ergebnis kombiniert werden, da sich vorhandene Ansätze zur Verbindung von verschiedenen Suchscores (vgl. z.B. Shanfeng et al.[54]) aufgrund einer anderen Ausgangssituation und Datenbasis nicht direkt auf die hier verwendeten Verfahren übertragen lassen, wird auch hier auf ein einfaches, intuitiv begründbares Verfahren zurückgegriffen: Die Punktzahl der Ähnlichkeitssuche wird über die Punktzahl der Schlüsselwortsuche abgewertet. Dazu wird die Punktzahl des direkten Schlüsselworttreffers, zu dem die Ähnlichkeit berechnet wurde mit dem Ergebnis der Ähnlichkeitsberechnung multipliziert. Im vorliegenden Beispiel ergab sich z.B. für D2 über die Schlüsselwortsuche ein Ergebnis von 0,5 (s.o.). Zu D2 werden nun die Dienste oder Operationen betrachtet, zu denen sich eine Ähnlichkeitsbeziehung berechnen lässt. So ist z.B. beträgt z.B. die Ähnlichkeit von D3 zu D2 0,53 (s. Tabelle 3.3). Das bedeutet also, dass D3 zu 53% mit einem Dienst übereinstimmt, der zu 50% den Suchbegriffen entspricht. Diese beiden Werte werden nun multipliziert, so dass sich für D3 über die Ähnlichkeit zu D2 ein Suchergebnis von 0,27 ergibt. Um auf diese Weise nicht zu viele, irrelevante Suchergebnisse zu produzieren muss die berechnete Punktzahl über einem Grenzwert liegen, der im Suchsystem festgelegt werden kann um ein gültiges Suchergebnis zu sein.

Es kann vorkommen, dass ein Dienst auf verschiedenen Wegen als Suchergebnis ermittelt werden kann. So wurde z.B. für D3 bereits eine Punktzahl von 0,27 über die Ähnlichkeit zu D2 berechnet. D3 verfügt zusätzlich auch über eine Ähnlichkeit von 0,27 zu D1. Da D1 ebenfalls ein Treffer der Schlüsselwortsuche mit einer Punktzahl von 1,0 ist, lässt sich für D3 auch über diesen alternativen Rechenweg ein Suchergebnis von 0,27 errechnen. Dass diese beiden Werte übereinstimmen, ist ein Zufall, der sich aus dem einfachen Beispiel ergibt. Dass dies nicht die Regel ist zeigt, sich z.B. am Knoten O1. Die Schlüsselwortsuche ergab für O1 ein Ergebnis von 0,5. Zusätzlich weist O1 aber auch eine Ähnlichkeit von 0,96 zu D1 und 0,48 zu D2 auf. Berechnet man nun die Punktzahl für O1 über D1, so ergibt sich eine Punktzahl von 0,96, über D2 beträgt sie 0,24. Tabelle 3.4 zeigt die im Beispiel über alle möglichen Rechenwege ermittelten Ergebnisse. Die erste Zeile („SWS“) enthält die Suchergebnisse der Schlüsselwortsuche, die darauf folgenden Zeilen jeweils die über die Ähnlichkeitssuche mit den Ergebnissen der Schlüsselwortsuche errechneten Werte (Zeilen „D1“, „D2“ und „O1“).

	D1	**D2**	**D3**	**D4**	**O1**	**O2**	**O3**	**O4**	**O5**
SWS	1	0,5	0	0	0,5	0	0	0	0
D1	1	0,5	0,27	0	0,96	0,96	0,46	0,25	0
D2	0,25	0,5	0,27	0	0,24	0,24	0,46	0,25	0,13
O1	0,42	0,21	0,11	0	0,5	0,4	0,19	0,11	0

Tabelle 3.4: Beispiel: Berechnete Suchergebnisse

Lässt sich ein Dienst oder eine Operation über verschiedene Wege als Suchergebnis

errechnen, so wird die höchste erreichbare Punktzahl für ein Ergebnis verwendet. Es kann z.B. durchaus vorkommen kann, dass ein Knoten, der eine sehr niedrige Punktzahl bei der Schlüsselwortsuche erzielt, eine große Ähnlichkeit zu einem sehr guten Treffer hat und somit besseres Suchergebnis darstellt, als es die Schlüsselwortsuche vermuten lässt. In Tabelle 3.4 ist demnach der Maximalwert jeder Spalte das endgültige Suchergebnis. Sind zwei Dienste bzw. Operationen identisch, führt das verwendete Multiplikationsverfahren dazu, dass die Punktzahl gleich bleibt, egal ob sie über den Ähnlichkeitswert oder die Schlüsselwortsuche berechnet wird. Je höher die endgültige Punktzahl, desto besser wird das Suchergebnis eingestuft. Tabelle 3.5 enthält die sortierten Suchergebnisse für die Suchanfrage „Schlüsselwort1 Schlüsselwort2“.

Rang	Ergebnis	Punktzahl
1	Dienst1	1,0
2	Operation1	0,96
3	Operation2	0,96
4	Dienst2	0,5
5	Operation3	0,46
6	Dienst3	0,27
7	Operation4	0,25
8	Operation5	0,13

Tabelle 3.5: Beispiel: Sortierte Ergebnisliste

D4 hat keine berechenbare Ähnlichkeit zu einem der direkten Suchergebnisse, daher ist es kein Teil der Ergebnisliste. Die zu D4 gehörende Operation O5 wird jedoch als Ergebnis geliefert, was zeigt, dass wie gefordert die Eigenschaften von Operationsbeschreibungen nicht auf ihre Dienste übertragen werden. Wie bereits angemerkt wurde, ist in einer realen Situation im Verhältnis zu den Diensten und Operationen mit mehr Schlüsselwörtern und dementsprechend weniger dichten Zusammenhängen zwischen den Knoten zu rechnen. Dies führt dazu, dass ein deutlich größerer Anteil der im Dienstgraphen enthaltenen Knoten nicht als Ergebnisse ermittelt würden. Des Weiteren lässt sich beobachten, dass z.B. O5 mit 0,13 Punkten ein vergleichsweise niedriges Ergebnis erreicht hat. Zu schlechte Ergebnisse sollten aus der Ergebnisliste gefiltert werden, wobei getestet werden müsste welche Mindesthöhe sinnvoll ist. Eine Erhöhung des Grenzwertes erhöht die Präzision des Suchalgorithmus, jedoch auch die Wahrscheinlichkeit, dass relevante Suchergebnisse entfernt werden.

Da Umfang und Vollständigkeit der durch die Benutzer erstellten Beschreibungen bei der tatsächlichen Verwendung eines solchen Systems gerade in der Anfangsphase nicht

vorausgesagt werden kann, wurde hier ein einfaches Suchsystem gewählt, dessen Schwerpunkt darauf liegt, aus wenigen Informationen eine große Menge an Suchergebnissen zu produzieren. Dies verringert die Präzision der Suchergebnisse, denn je mehr Suchergebnisse insgesamt geliefert werden, desto größer ist die Wahrscheinlichkeit, dass irrelevante Ergebnisse enthalten sind. Wenn z.B. sehr allgemeine Begriffe für die Beschreibung vieler verschiedener Objekte verwendet werden, könnten diese über die Ähnlichkeitssuchfunktion fälschlicherweise als korrekte Ergebnisse erkannt werden. Aus diesem Grund wird über das Relevanz-Feedback die Entscheidung über die Relevanz der von der Suche ermittelten Ergebnisse langfristig an die Benutzergemeinschaft übergeben.

3.2.3 Relevanz-Feedback und Personalisiertes Suchranking

Die Aufgabe von Such- und Empfehlungsalgorithmen ist es, das Verhalten eines menschlichen Benutzers bei der Wahl zwischen verschiedenen Objekten zu imitieren. Die endgültige Entscheidung, ob ein ermitteltes Suchergebnis den Anforderungen entspricht trifft der Benutzer, der die von der Suchmaschine gelieferten Vorschläge betrachtet und den geeignetsten auswählt. Um die möglicherweise geringe Präzision des im vorherigen Abschnitt vorgestellten Verfahrens auszugleichen, verwendet das vorgestellte Suchsystem das in Kapitel 2.6.2 beschriebene Prinzip des Relevanz-Feedbacks zur endgültigen Sortierung der Suchergebnisse. Der Benutzer hat nach jeder Suche die Möglichkeit, Ergebnisse als relevant oder nicht relevant zu bewerten. Das Suchsystem merkt sich, ob das bewertete Ergebnis für die vorliegenden Suchbegriffe als relevant oder nicht relevant beurteilt worden ist. Bei der nächsten Verwendung dieser Suchbegriffe wirken sich die vergangenen Bewertungen positiv bzw. negativ auf die Punktzahl und somit auf die Position in der Ergebnisliste aus. Die Benutzer sind damit nicht nur für die Dokumentation der Dienste verantwortlich, sondern bestimmen auch langfristig durch ihre Bewertungen die Sortierung der Suchergebnisse. Da nicht jeder Benutzer dieselbe Einschätzung hat, ob ein Ergebnis relevant ist oder nicht, werden die Bewertungen an den Inhalt von Benutzerprofile gebunden und somit eine Personalisierung erreicht. Die Funktionsweise dieses Systems wird im Folgenden erläutert.

Als Basis für die Personalisierung sollen Benutzerprofile verwendet werden. Dabei soll es sich um wissensbasierte, vom Benutzer ausgefüllte Profile handeln, und nicht wie es bei Empfehlungssystemen häufig der Fall ist, um automatisch durch Beobachtung des Nutzungsverhaltens erstellte(vgl. Middleton et al.[38]). Die Grundidee für Aufbau und Verwendung des Profils stammt aus der Arbeit von Kuck et al.[33] zum kontextsensitiven Service Discovery. Dort besteht ein Benutzerprofil aus einer Menge von Schlüsselwörtern, die den Kontext des Benutzers beschreiben. Diese werden mit einer Suchanfrage übergeben

und zur Ermittlung von personalisierten, an den Benutzerkontext angepassten Ergebnissen verwendet. Ein vergleichbares Prinzip wird hier angewendet. Ein Benutzerprofil enthält Schlüsselwörter, die Eigenschaften des Nutzers beschreiben können. Um welche Eigenschaften es sich dabei genau handelt, soll hier vorerst nicht genau festgelegt werden, da sich die relevanten Informationen abhängig von ihrer Domäne unterscheiden können, in einem geschäftlichen Umfeld könnte es z.B. sinnvoll sein Eigenschaften wie „Branche“, „Abteilung“ oder „Position“ vorzugeben, um eine Gruppierung der Nutzer zu ermöglichen. Die Profileinträge sind dabei Schlüssel-/Wert-Paare, wobei der Schlüssel die Bezeichnung der Eigenschaft darstellt. Im Rahmen der Personalisierung werden die Werte der Eigenschaften als Schlagworte zur Beschreibung des Benutzers verwendet. Gibt ein Benutzer eine Relevanz-Bewertung ab, werden diese Benutzerbegriffe der Bewertung zugeordnet um nachvollziehen zu können, welche Eigenschaften Benutzer haben, die ein bestimmtes Ergebnis als relevant bzw. nicht relevant bewerten. Führt nun ein anderer Benutzer eine Suche aus, werden dessen Benutzerbegriffe übergeben und mit zuvor abgegebenen Relevanzbewertungen verglichen. Bewertungen von Nutzern mit ähnlichen Profilen werden höher gewichtet, als die von Benutzern deren Profil keine Übereinstimmungen aufweist, somit erhalten, wenn genügend Bewertungen vorliegen, Benutzer mit unterschiedlichen Profilen eine unterschiedlich sortierte Ergebnisliste. Auf den ersten Blick ähnelt dieses Vorgehen dem in Kapitel 2.6.1 vorgestellten „Collaborative Filtering“. Auch dort geben Benutzer Bewertungen ab und erhalten auf Basis der Bewertungen ähnlicher Benutzer Empfehlungen (vgl. Mobasher[39]). Der Unterschied liegt jedoch darin, dass die Bewertungen beim hier verwendeten Vorgehen nicht an individuelle Benutzer, sondern an Begriffe aus den Benutzerprofilen gebunden werden. Dies führt dazu, dass sich eine Änderung des Benutzerprofils unmittelbar auf die gelieferten Suchergebnisse auswirkt.

Die genaue Funktionsweise der Relevanzbewertungen soll an dem bereits bekannten Beispiel aus Kapitel 3.2.2 beschrieben werden (vgl. Tabelle 3.2). Zur Demonstration der Personalisierung werden die in Tabelle 3.6 dargestellten Benutzerprofile verwendet, die zur Vereinfachung nur über eine Benutzereigenschaft verfügt.

Benutzername	**Eigenschaft**
Benutzer1	Benutzerbegriff1
Benutzer2	Benutzerbegriff1
Benutzer3	Benutzerbegriff2
Benutzer4	Benutzerbegriff2

Tabelle 3.6: Beispiel: Einfache Benutzerprofile

Wird eine Relevanzbewertung abgegeben, merkt sich das System sämtliche Suchbegriffe der Suchanfrage die zum bewerteten Ergebnis geführt hat. Zu jedem dieser Suchworte

werden nun die Benutzerbegriffe aus dem Profil des bewertenden Benutzers hinzugefügt, mit einer Zählung der „Ja“- bzw. „Nein“-Stimmen. Da sich die Zahl der Such- und Benutzerbegriffe unterscheiden kann, muss außerdem die Gesamtzahl der Stimmen erfasst werden, um auch dann eine Relevanz-Vorhersage treffen zu können, wenn es noch keine Bewertungen von Nutzern mit Übereinstimmungen im Profil gibt. Nun führen Benutzer1, Benutzer2 und Benutzer3 die Suchanfrage „Schlüsselwort1 Schlüsselwort2“ aus, die ohne Berücksichtigung von Relevanzfeedback die in Tabelle 3.5 gezeigten Ergebnisse liefert. Benutzer4 sucht nur nach „Schlüsselwort1“. Daraufhin bewerten alle vier Benutzer D1, O1 und O2 wie in in Tabelle 3.7 dargestellt. Ein Plus steht dabei jeweils für eine positive, ein Minus für eine negative Relevanzbewertung. Ein leeres Feld bedeutet, dass keine Bewertung für dieses Ergebnis abgegeben wurde.

	D1	O1	O2
Benutzer1	+	+	+
Benutzer2	+	-	
Benutzer3	-	+	
Benutzer4	-	-	

Tabelle 3.7: Beispiel: Relevanzbewertungen

Ergebnis	Suchbegriff	Benutzerbegriff	Ja	Nein
D1	**Schlüsselwort1**	**Gesamtstimmen**	**2**	**2**
		Benutzerbegriff1	2	0
		Benutzerbegriff2	0	2
	Schlüsselwort2	**Gesamtstimmen**	**2**	**1**
		Benutzerbegriff1	2	0
		Benutzerbegriff2	0	1
O1	**Schlüsselwort1**	**Gesamtstimmen**	**2**	**2**
		Benutzerbegriff1	1	1
		Benutzerbegriff2	1	1
	Schlüsselwort2	**Gesamtstimmen**	**2**	**1**
		Benutzerbegriff1	1	1
		Benutzerbegriff2	1	0
O2	**Schlüsselwort1**	**Gesamtstimmen**	**1**	**0**
		Benutzerbegriff1	1	0
	Schlüssewort2	**Gesamtstimmen**	**1**	**0**
		Benutzerbegriff1	1	0

Tabelle 3.8: Beispiel: Erfassung der Relevanzbewertungen

Tabelle 3.8 zeigt, wie diese Bewertungen vom System erfasst werden. Wie zu sehen ist, findet keine Zuordnung der Bewertungen zu individuellen Benutzern statt, es werden lediglich die Begriffe aus den Benutzerprofilen der abstimmenden Benutzer erfasst. Das bedeutet, dass eine Änderung am Benutzerprofil unmittelbar zu anderen Suchergebnissen führen kann. Für O2 hat nur Benutzer1 eine Bewertung abgegeben, entsprechend gibt es keine Stimmen mit Benutzerbegriff2. Da Benutzer4 nur nach Schlüsselwort1 gesucht hat, wird seine Stimme im Gegensatz zu den Stimmen der anderen Benutzer auch nur für dieses Schlüsselwort gezählt.

Zur Berechnung des Relevanzwertes für ein Suchergebnis wird zu jedem bei der Suche verwendeten Suchbegriff nach Relevanzbewertungen gesucht, die Benutzerbegriffen zugeordnet sind, die auch im Profil des suchenden Nutzers vorkommen. Aus diesen Werten wird der Durchschnitt berechnet und als personalisierter Relevanzwert für den entsprechenden Suchbegriff verwendet. Der Durchschnitt dieser Relevanzwerte für die einzelnen Suchbegriffe wird nun als Relevanzvorhersage für das Ergebnis zu den vorliegenden Suchbegriffen verwendet. Die Einschätzung einzelner Benutzer kann von der durchschnittlichen Meinung aller Nutzer abweichen. Da die Anzahl der Benutzer mit Überschneidungen im Profil geringer ist, als die Gesamtzahl der Benutzer, wirken sich derartige Ausreißer stärker auf das Ergebnis der Berechnung aus, als wenn auf eine Personalisierung verzichtet würde. Um diesen Effekt abzuschwächen, besteht die Möglichkeit, das Ergebnis mit Hilfe der nicht personalisierten Relevanzvorhersage zu normalisieren. Dies wird über die Verwendung eines Personalisierungsfaktors erreicht, der aussagt, wie stark sich das personalisierte Ergebnis auf das Gesamtergebnis auswirken soll. Der endgültige Relevanzwert errechnet sich daher als der, anhand des Personalisierungsfaktors gewichtete Durchschnitt, aus der personalisierten Relevanzbewertung und dem Durchschnitt sämtlicher abgegebener Bewertungen. Der Wert des Faktors liegt zwischen „0“ und „1“, wobei zwischen „0“ und „0,5“ die nicht personalisierte, zwischen „0,5“ und „1“ die personalisierte Relevanz stärker gewichtet wird. Welcher Personalisierungsfaktor die zufriedenstellendsten Ergebnisse liefert, ist schwer vorherzusagen und müsste in einem längerfristigen Test ermittelt werden.

In der Arbeit von Shanfeng et al.[54] wirkt sich das Ergebnis der Bewertung direkt auf die Punktzahl eines Suchergebnisses, sowie die Punktzahlen von ähnlichen Ergebnissen aus. Dieses Vorgehen würde im vorliegenden Fall der Anforderung widersprechen, dass die endgültige Entscheidung, ob ein Suchergebnis relevant ist beim Benutzer liegt. So wäre es beispielsweise denkbar, dass aufgrund einer unvollständigen Beschreibung oder ungünstig gewählter Suchbegriffe ein Ergebnis vom Suchalgorithmus als sehr niedrig eingestuft wird, dass aber genau dem entspricht, wonach der Benutzer gesucht hat. Würde die Relevanzpunktzahl mit der Suchpunktzahl kombiniert, so wären mehr Stimmen notwendig,

um dem Ergebnis eine aussagekräftige Punktzahl zuzuweisen, als dies der Fall wäre, wenn die Beschreibung besser wäre. Auch die Übertragung der Relevanz auf ähnliche Ergebnisse würde voraussetzen, dass der zugrunde liegende Suchalgorithmus sicher zutreffende Ergebnisse liefert. Genau dies kann aber aus den in den vorherigen Abschnitten erwähnten Gründen nicht garantiert werden, was letztendlich eine Begründung für die Verwendung des Relevanzfeedbacks lieferte. Wenn also ein Ergebnis relevant ist und ein laut Suchalgorithmus sehr ähnliches Ergebnis nicht, dann würden sich die Relevanzbewertungen der beiden Ergebnisse aufgrund der Ähnlichkeit teilweise aufheben. Aufgrund dieser Probleme mit der Zusammenfassung von Such- und Relevanz-Punktzahl wird auf ein derartiges Verfahren verzichtet. Stattdessen wird mit zwei getrennten Werten gerechnet, wobei die Relevanz-Punktzahl als primäres Sortierkriterium für die Ergebnisliste genutzt wird, während die Suchpunktzahl bei Ergebnissen mit gleicher Relevanzbewertung, also z.B. vor der ersten Stimmabgabe für die Sortierung verwendet wird.

Das Ziel ist es, eine Sortierung zu erreichen, in der die bereits von Benutzern für relevant befundenen Ergebnisse zuerst vorkommen, gefolgt von Ergebnissen für die bisher keine Bewertung vorliegt und den Ergebnissen die bereits als nicht relevant bewertet worden sind am Ende. Dazu wird für alle Suchergebnisse vor Abgabe der ersten Bewertungen ein Startwert von „0,5“ vorgegeben, der eine neutrale Relevanzbewertung darstellen soll. Dies wird erreicht, indem jedem Ergebnis als Ausgangswert jeweils eine halbe „Ja“- und eine halbe „Nein“-Stimme, also eine bei normaler Stimmabgabe nicht mögliche neutrale Stimme. Je mehr reale Bewertungen abgegeben werden, desto weniger fällt diese neutrale Stimme ins Gewicht. Dieses Vorgehen bietet gegenüber der Alternative, den neutralen Startwert nicht als eigene Stimme zu werten den Vorteil, dass verhindert wird, dass ein Ergebnis nach nur einer Stimme eine rechnerische Relevanz von 100% bzw. 0% erhält und sich somit direkt an die Spitze bzw. das Ende der Ergebnisliste setzt. Nach Abgabe von einzelnen Stimmen ist das Ergebnis noch nicht so repräsentativ, wie nach mehrfacher Stimmabgabe, durch die neutrale Stimme werden die ersten abgegebenen Stimmen abgewertet. Erhält ein Ergebnis beispielsweise eine erste, positive Bewertung, so liegt die errechnete Relevanz bei „0,75“, anstatt der „1,0“, die ohne die neutrale Stimme erreicht würde. Nach einer weiteren bei ca. „0,83“ und nach der dritten positiven Stimme bei ca. „0,88“. Das Erreichen von „1,0“ bzw. „0,0“ mit nur wenigen Stimmen wird verhindert, auch wenn bei der verwendeten Berechnung über einen Durchschnittswert nach wie vor das grundsätzliche Problem besteht, dass Bewertungen ein höheres Gewicht haben, je weniger es insgesamt gibt. Die vom vorgestellten Suchalgorithmus errechnete Punktzahl dient nach der Abgabe von Relevanzbewertungen nur noch als sekundäres Sortierkriterium, für Ergebnisse, die zufällig dieselbe Relevanz haben, insbesondere aber für alle Ergebnisse, die bislang keine Bewertungen erhalten haben und somit eine neutrale Relevanz von „0,5“ besitzen.

Zur Erläuterung des Verfahrens wird nun angenommen, dass Benutzer1 und Benutzer3, nachdem sämtliche in Tabelle 3.7 angegebenen Relevanzbewertungen abgegeben wurden, erneut die Suchanfrage „Suchbegriff1 Suchbegriff2“ ausführen. Da die anderen beiden Benutzer mit Benutzer1 bzw. Benutzer3 identische Benutzerprofile haben, bekommen sie auf dieselben Suchanfragen auch dieselben Ergebnisse, werden deshalb nicht getrennt betrachtet. Für D1 sind aus Sicht von Benutzer1 zwei positive Stimmen von Benutzern mit gleichem Profil und zwei negative Stimmen von Benutzern ohne Übereinstimmungen im Profil abgegeben worden. Unter Einbeziehung der oben vorgestellten ersten neutralen Stimme ergibt sich für Benutzer1 für die beiden verwendeten Suchbegriffe jeweils eine personalisierte Relevanzvorhersage von 0,83. Insgesamt wurden für Suchbegriff1 zwei positive und zwei negative Stimmen abgegeben, was wieder inklusive einer neutralen Stimme eine Gesamtbewertung von 0,5 ergibt. Da Suchbegriff2 eine negative Stimme weniger bekommen hat beträgt, die Gesamtbewertung für Suchbegriff2 0,63. Für jeden Suchbegriff wird nun der anhand des Personalisierungsfaktors gewichtete Durchschnitt zwischen der nicht personalisierten Gesamtbewertung, also dem Durchschnitt der Relevanzbewertungen sämtlicher Benutzer, und der personalisierten Bewertung berechnet. Wählt man z.B. einen Faktor von 4/5, der ein stark personalisiertes Ergebnis liefert, so ergibt sich für Suchbegriff1 ein Wert von 0,77, für Suchbegriff2 0,79. Aus den Werten aller Suchbegriffe wird nun der Durchschnitt gebildet, so dass eine Relevanzvorhersage von 0,78 entsteht.

Für Benutzer3 gibt es für Schlüsselwort1 zwei und für Schlüsselwort2 eine negative Stimme von einem Benutzer mit übereinstimmenden Profil. Daraus ergibt sich insgesamt eine Relevanzvorhersage von 0,28. Ein niedriger gewählter Personalisierungsfaktor würde dazu führen, dass sich die Werte für Benutzer1 und Benutzer3 einander nähern.

Personalisierte Suchergebnisse für Benutzer1 und Benutzer2			
Position	Suchergebnis	Relevanzvorhersage	Suchpunktzahl
1	D1	0,78	1,0
2	O2	0,75	0,96
3	O1	0,51	0,96
4	D2	0,5	0,5
5	O3	0,5	0,46
6	D3	0,5	0,27
7	O4	0,5	0,25
8	O5	0,5	0,13

Tabelle 3.9: Beispiel: Suchergebnisse für Benutzer1/Benutzer2

O2 hat nur eine Stimme von Benutzer1 bekommen. Da in diesem Fall die Gesamtzahl der Stimmen der von Benutzern mit Benutzerbegriff1 im Profil entspricht, findet keine Personalisierung statt. Für Benutzer1 und Benutzer2 ist der personalisierte Wert gleich dem Gesamtwert, für die anderen Benutzer wird nur der Gesamtwert betrachtet, so dass sich für alle Benutzer eine Relevanzvorhersage von 0,75 ergibt. Die Tabellen 3.9 und 3.10 zeigen die endgültige Sortierung der Ergebnisse der Suchanfrage „Schlüsselwort1 Schlüsselwort2“ für Benutzer1/Benutzer2 und Benutzer3/Benutzer4. Die drei bewerteten Ergebnisse sind rot markiert, die unbewerteten Ergebnisse werden anhand ihrer zuvor berechneten Suchpunktzahl sortiert.

Personalisierte Suchergebnisse für Benutzer3 und Benutzer4			
Position	Suchergebnis	Relevanzvorhersage	Suchpunktzahl
1	O2	0,75	0,96
2	O1	0,61	0,96
3	D2	0,5	0,5
4	O3	0,5	0,46
5	D3	0,5	0,27
6	O4	0,5	0,25
7	O5	0,5	0,13
8	D1	0,28	1,0

Tabelle 3.10: Beispiel: Suchergebnisse für Benutzer3/Benutzer4

3.3 Zusammenfassung und Beurteilung

Ziel der Arbeit ist die Entwicklung eines Suchsystems, das benutzergenerierte Beschreibungen von Web Services verwendet, um die Schwächen vorhandener Web Service Suchmechanismen zu beheben. Es gibt verschiedene Ansätze aus den Bereichen des Service Discovery und Information Retrieval, die sich mit der Realisierung und Optimierung von Suchalgorithmen beschäftigen. Eine direkte Übertragung dieser Verfahren ließ sich aufgrund der folgenden Besonderheiten der vorliegenden, durch eine Benutzercommunity generierten Datenbasis nicht rechtfertigen:

- Umfang, Korrektheit und Vollständigkeit der vorliegenden Informationen lässt sich gerade in der Anfangsphase des Systems nicht vorhersagen, und auch langfristig ist anzunehmen, dass die verwendeten Beschreibungen nur aus wenigen Sätzen bestehen, was eine Gewichtung von Suchbegriffen und die Einschätzung der Qualität automatischer Suchalgorithmen erschwert.

- Im Gegensatz zu vielen anderen Webservice-Suchsystemen sollen nicht nur Dienste, sondern auch Operationen als Suchergebnisse in Frage kommen, was die Berücksichtigung der Hierarchie der vorliegenden Daten voraussetzt. Dies wird nur von wenigen Suchalgorithmen unterstützt.

Aus diesem Grund wurden zur Realisierung der Suche einfache Algorithmen gewählt, deren Ziel es ist, aus einer möglicherweise geringen Datenmenge eine große Zahl an Ergebnissen zu erzeugen, ohne dabei die aus dem hierarchischem Aufbau der vorhandenen Daten entstandenen Anforderungen zu verletzen. Die zur Suche verwendeten Daten stammen vom Benutzer und so sollte auch die endgültige Entscheidung darüber, ob ein Suchergebnis relevant ist oder nicht beim Benutzer liegen. Das Suchsystem verfügt daher über ein Feedback-System, das Bewertungen durch die Benutzer sammelt und die Suchergebnisse so langfristig eher den Bedürfnissen der Benutzer entspricht, als die verwendeten Suchalgorithmen dies aufgrund der zuvor genannten Einschränkungen leisten können. Um die Qualität der Suchergebnisse zusätzlich zu steigern, wird das Relevanzfeedback zusätzlich verwendet, um die Suchergebnisse anhand von Benutzerprofilen zu personalisieren.

4 Implementierung

Nachdem im vorherigen Kapitel die Anforderungen sowie die verwendeten Verfahren und Algorithmen vorgestellt wurden, beschäftigt sich dieses Kapitel mit der technischen Umsetzung des Suchsystems. Im ersten Unterkapitel wird SiSeOr vorgestellt, eine Designumgebung, die als Basis für das neu entwickelte Suchsystem dient. Als nächstes werden die Datenhaltung, die einzelnen Komponenten des Suchsystems und die Umsetzung der Benutzeroberfläche beschrieben.

4.1 Die SiSeOr-Orchestrierungsumgebung

Das im letzten Kapitel vorgestellte Web Service Suchsystem soll als Teil der SiSeOr-Orchestrierungsumgebung implementiert werden. SiSeOr baut auf der EUSOP-Plattform auf, die im folgenden Abschnitt vorgestellt wird.

4.1.1 EUSOP

Wie bereits in Kapitel 2.1.1 festgestellt wurde, sind vorhandene Systeme zur Orchestrierung von Web Services zu komplex, um von Endbenutzern verwendet zu werden. EUSOP (End User Service Orchestration Plattform) wurde entwickelt, um Endbenutzern die eigenständige Serviceorchestrierung zu ermöglichen, indem sie beim Auffinden, Verstehen, sowie bei der Nutzung und Komposition von Diensten unterstützt werden (vgl. Hofmann et al. [26]). Um dieses Ziel zu erreichen, werden durch die Benutzer-Community generierte Metadaten wie z.B. Beschreibungstexte, Kommentare, Tags und Bewertungen verwendet. Die Speicherung dieser Metadaten erfolgt im UDDI-Verzeichnis und als Dokumentation innerhalb der WSDL-Dienstbeschreibung. Die normalerweise im UDDI enthaltenen Informationen zur Beschreibung von Diensten werden von EUSOP um zusätzliche Metadaten erweitert. UDDI ist als Dienstverzeichnis für die Suche nach Diensten verantwortlich, deshalb werden im UDDI solche Informationen gespeichert, die für einen Benutzer bei der Suche nach einem Webservice von Interesse sein könnten. Im einzelnen können die UDDI-Beschreibungen der Dienste um folgende Informationen erweitert werden:

- Beschreibungstexte
- Autoren

- Versionsnummern
- Erstellungsdatum
- Tags
- Bewertungen

WSDL-Dateien stellen die technische Beschreibung von Web Services dar. Diese werden nun um Nicht-funktionale, von den Benutzern erstellten Metadaten erweitert. Um aus den WSDL-Dateien vollständige Beschreibungen der Dienste zu machen, werden sie von EUSOP um alle Daten ergänzt, die von EUSOP bereits im UDDI gespeichert werden, sowie zusätzliche Informationen, die für eine Suche nach Webservices nicht relevant sind und deshalb im UDDI keine sinnvolle Funktion erfüllen würden. Die Metadaten werden in die „<documentation>“-Elemente (vgl. Kapitel 2.1.1) der WSDL-Dateien eingetragen. Diese Elemente können für beliebige Informationen in Form natürlichsprachlicher Texte verwendet werden. In EUSOP werden sie als Ablage für klar definierte Metadaten verwendet, die gezielt ausgelesen werden können. Die von EUSOP in den WSDL-Dateien gespeicherten Metadaten umfassen zusätzlich zu den bereits im UDDI enthaltenen Daten:

- Kommentare
- Referenzen (z.B. Webseite des Anbieters, externe Dokumentation des Dienstes)
- Operations- und Attributbeschreibungen
- Kategorien

Neben diesen fest definierten Metadaten bietet EUSOP die Möglichkeit, frei definierbare Schlüssel-/Wert-Paare im WSDL abzuspeichern, um eine einfache Erweiterung um zusätzliche Informationen zu ermöglichen. Tags werden in EUSOP den Diensten zugeteilt und sind eindeutig. Eine Zuordnung zum Benutzer, der ein Schlagwort eingegeben hat erfolgt nicht. Dies verhindert die Zählung und Gewichtung von Tags anhand der Häufigkeit ihrer Nutzung durch unterschiedliche Benutzer. Da WSDL-Dateien i.d.R nicht lokal vorliegen, und entsprechend nicht verändert werden können, geschieht die Speicherung in Form von XSLT-Transformationsregeln[69]. Abbildung 4.1 zeigt den Ablauf Transformation und Verwendung von WSDL-Dateien in EUSOP. Innerhalb von EUSOP werden die Metadaten in Form des Service-Objekts verwendet. Dieses kann aus einer WSDL-Datei generiert werden (1) und enthält sämtliche bereits in der ausgelesenen WSDL-Datei vorhandenen Metadaten. Werden die Metadaten im Service-Objekt verändert, so kann EUSOP daraus eine XSLT-Datei generieren (2). Diese enthält XSLT-Transformationsregeln die dazu genutzt werden

können, aus der Original-WSDL-Datei eine neue, um die zuvor im Service-Objekt veränderten oder hinzugefügten Metadaten erweiterte WSDL-Datei zu generieren (3). Aus so dokumentierten WSDL-Dateien kann EUSOP wiederum die Metadaten auslesen und ein Service-Objekt erzeugen(1).

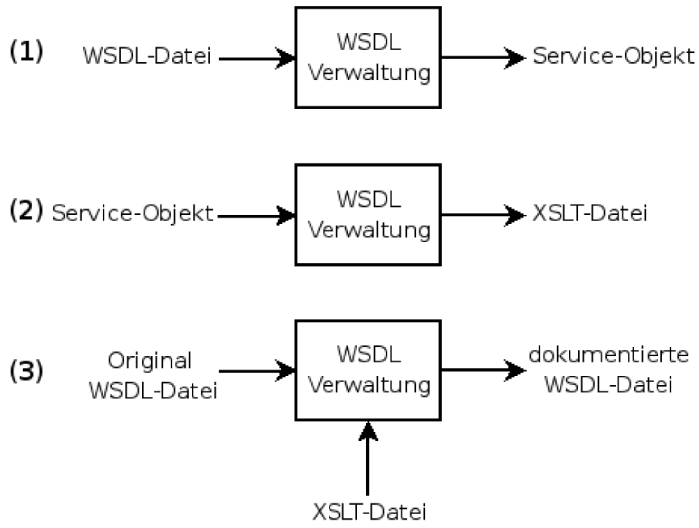

Abbildung 4.1: WSDL-Verarbeitung in EUSOP

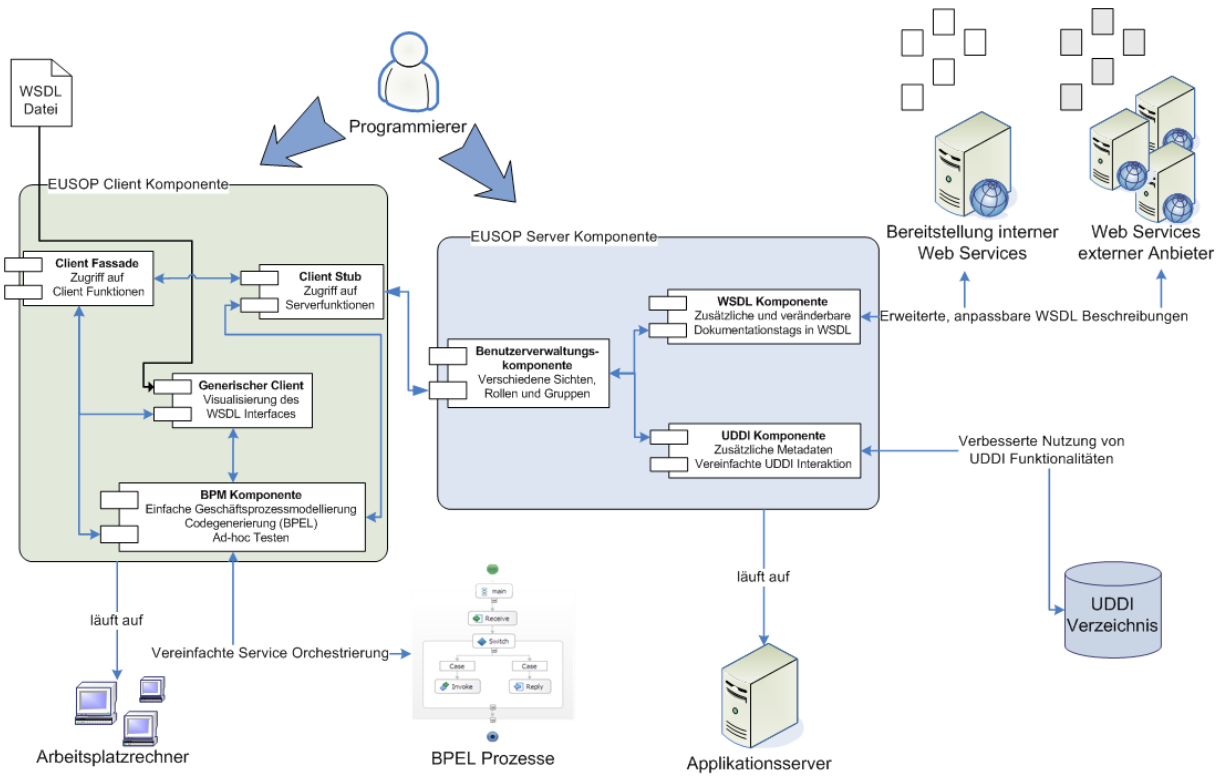

Abbildung 4.2: EUSOP – End User Service Orchestration Platform [26]

Abbildung 4.2 zeigt die Architektur von EUSOP. Das System besteht aus einer Client- und

einer Server-Komponente. Die Server-Komponente ist für den Zugriff auf die Dienstbeschreibungen im WSDL und UDDI zuständig, während der Client die Elemente der Dienstbeschreibung interpretiert und Java-Stellvertreterobjekte für die Web Services erstellt. Diese Objekte enthalten alle Informationen über die Schnittstellen der Dienste, die zur Orchestrierung benötigt werden. Eine weitere Funktion der Client-Komponente ist die Generierung von BPEL-Code aus einer mit diesen Stellvertreterobjekten erstellten Dienstkomposition (vgl. Hofmann et al. [26]).

4.1.2 SiSeOr

Der EUSOP-Client bietet eine Schnittstelle für graphische Orchestrierungseditoren, die das dort verwendete Datenmodell sowie die im EUSOP-Server gespeicherten Metadaten nutzen können. SiSeOr (Simple Service Orchestration) ist ein auf der EUSOP-Plattform aufbauender, grafischer Orchestrierungseditor, der die „Box and Wire“-Metapher verwendet. Operationen werden als Rechtecke dargestellt, die Datenflüsse zwischen den Operationen als gerichtete Verbindungslinien (vgl. Paczynski [45]).

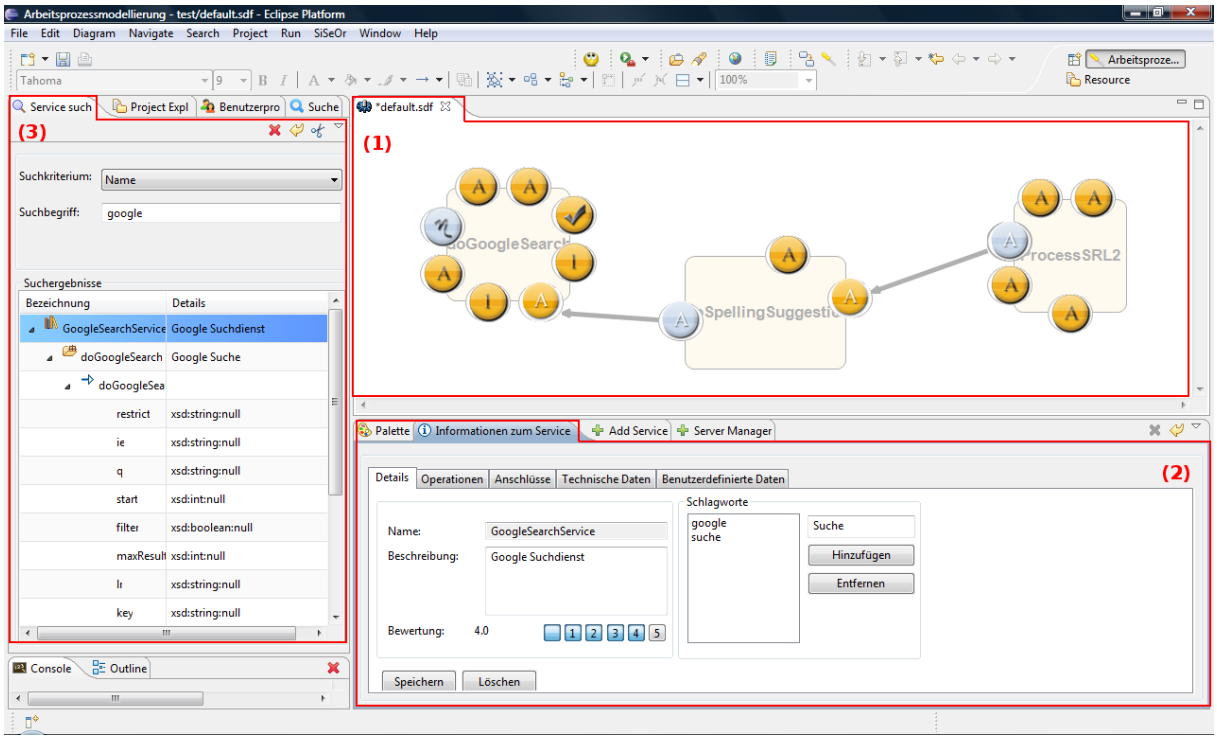

Abbildung 4.3: Die SiSeOr-Benutzeroberfläche

Abbildung 4.3 zeigt die Umsetzung in der SiSeOr-Oberfläche. Rechts oben sieht man den eigentlichen Orchestrierungseditor (1), darunter die vom Benutzer veränderbaren Felder zur Eingabe der Metadaten (2), die dann von EUSOP im UDDI bzw. WSDL gespeichert werden. SiSeOr bietet zur Zeit eine einfache, auf den im UDDI gespeicherten Metadaten aufbauende

Suchfunktion, die man auf der linken Seite der Abbildung erkennt (3). Der Benutzer wählt aus, nach welchen Informationen er suchen will, gibt einen Suchbegriff ein und erhält das Ergebnis in Form einer Baum-Struktur, die gefundene Dienste und alle ihre Operationen und Attribute enthält. Wählt der Benutzer nun eine Operation aus, so wird diese in die Palette übernommen, die sich im selben Unterfenster wie die Service-Informationen findet (2). Die Palette enthält Symbole für die Operationen, die der Benutzer gerne verwenden möchte. Diese können dann als „Bausteine“ in den Orchestrierungseditor (1) übernommen werden.

An der Suchfunktion zeigt sich die Ausrichtung von EUSOP auf die Dienst-Ebene, eine Suche nach Operationen ist nicht möglich. Stattdessen muss der Benutzer innerhalb der gelieferten Suchergebnisse von Hand nach den passenden Operationen suchen, um diese zur Orchestrierung verwenden zu können. Die oben genannten Metadaten beziehen sich größtenteils auf die Dienste, die einzige Möglichkeit Operationen zu beschreiben sind natürlichsprachliche Beschreibungstexte aus den WSDL-Dateien, die nicht in die bisherige, UDDI-basierte Suchfunktion von SiSeOr einbezogen werden. Auch für das neue Suchsystem ergibt sich hieraus die Schwierigkeit, dass Tags und Kategorien nur für Dienste zur Verfügung stehen, zur Beschreibung von Operationen können ausschließlich aus den Beschreibungstexten extrahierte Schlüsselwörter verwendet werden.

SiSeOr bietet bislang keine direkte Unterstützung für die Verwendung von bestehenden Kompositionen als Komponenten neuer Kompositionen. Zwar lässt sich eine Komposition, solange sie über eine Web Service Schnittstelle verfügt, zur Komposition verwenden, es lässt sich aber keine Verbindung zwischen einer Komposition und den von ihr verwendeten Diensten herstellen. In Kapitel 3.2.1 wurde bereits ein vereinfachtes Modell zur Berücksichtigung von Kompositionen vorgestellt, dass sich trotz dieser Einschränkung anwenden lässt. Zu diesem Zweck wurden die von EUSOP genutzten Metadaten im Rahmen dieser Arbeit um ein neues Feld erweitert, dass Verweise auf Kompositionen enthält, die auf einen Dienst zugreifen. EUSOP verfügt grundsätzlich über ein System zur Benutzer- und Rollenverwaltung (vgl. Hofmann et al.[26]), dies ist jedoch bislang nicht vollständig implementiert und an SiSeOr angebunden, weshalb es sich nicht in Verbindung mit den hier genutzten Benutzerprofilen verwenden lässt. SiSeOr ist eine auf der Eclipse Rich Client Platform (RCP) basierende Anwendung. Diese Plattform wird im folgenden Abschnitt vorgestellt.

4.1.3 Eclipse Rich Client Platform

Die Eclipse RCP ist eine Plattform zur Entwicklung von Anwendungen in Java. Sie bietet im Grunde eine leere Benutzeroberfläche, die über Plugins mit beliebigen Inhalten und

Funktionen gefüllt werden kann (vgl. McAffer u. Lemieux[36]). Das bekannteste RCP-Plugin ist die Eclipse IDE (Integrated Development Environment), eine weit verbreitete und beliebte Entwicklungsumgebung für Java, die selbst wiederum mit Hilfe von weiteren Plugins um zusätzliche Funktionen wie z.B. die Unterstützung für weitere Programmiersprachen erweitert werden kann. Der Vorteil bei der Verwendung der Eclipse RCP ist, dass sich der Entwickler auf seine eigentliche Aufgabe konzentrieren kann, anstatt sich Gedanken über die Infrastruktur zu machen zu müssen, da die RCP diese zur Verfügung stellt. Sämtliche auf RCP basierenden Anwendungen werden in Form von Eclipse-Plugins entwickelt, die dann mit Hilfe der RCP zu einer Gesamtanwendung zusammengefügt werden.

Eine Eclipse-Benutzeroberfläche besteht aus der „Workbench“, einer Sammlung von Unterfenstern („Views“), in denen in Registerkarten ähnliche Fenster gruppiert werden können. Eclipse bietet die Möglichkeit so genannte Perspektiven zu erstellen. Diese stellen Anordnungen Views dar, die einem gemeinsamen Zweck dienen. Je nachdem welche Aufgabe ein Benutzer gerade ausführt, kann er zwischen verschiedenen Perspektiven wechseln. Im Falle der Eclipse Java IDE besteht z.B. die Möglichkeit zwischen einer Debugging- und einer Java-Entwicklungs-Perspektive zu ändern, die unterschiedliche Views sowie ein anderes Fenster-Layout verwenden. Der Inhalt der einzelner Views wird bei Eclipse mit Hilfe von SWT (Standard Widget Toolkit) und JFace entwickelt. Diese Bibliotheken bieten plattformunabhängige Komponenten, zum Aufbau von Benutzeroberflächen(vgl. McAffer u. Lemieux [36]). Abbildung 4.3 zeigt am Beispiel von SiSeOr die Oberfläche einer Eclipse RCP-Anwendung.

4.1.4 Anbindung des Suchsystems

Da es sich bei SiSeOr um eine Eclipse-Anwendung handelt, wird auch das Suchsystem als Eclipse-Plugin entwickelt. Die Benutzeroberfläche besteht aus zwei Views, für die Benutzerverwaltung und die eigentliche Suchfunktion, die in die SiSeOr-Oberfläche eingebunden werden. Abbildung 4.4 zeigt die Anbindung des Suchsystems an die EUSOP-/ SiSeOr-Plattform und die Kommunikation zwischen den Teilsystemen.

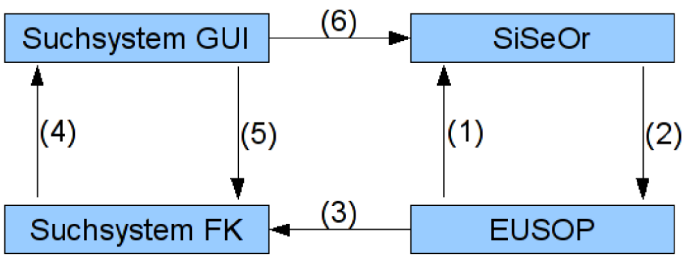

Abbildung 4.4: Anbindung Suchsystem an EUSOP/SiSeOr

SiSeOr holt sich die Metadaten zu einem Dienst aus EUSOP (1). Werden vom Benutzer Änderungen an diesen Metadaten vorgenommen oder auch ein neuer Dienst hinzugefügt, so werden diese an EUSOP übergeben und dort im UDDI und WSDL gespeichert (2). Wenn dies passiert, benachrichtigt EUSOP das Suchsystem (3), dass dann die veränderten Metadaten in den eigenen Datenbestand übernimmt (vgl. Kapitel 4.2.1) und die Ähnlichkeiten neu berechnet. Beim neuen Suchsystem handelt es sich um ein Plugin, das bedeutet, dass SiSeOr auch weiterhin funktionieren muss, wenn es nicht vorhanden ist. Um die Benachrichtigung vom EUSOP zum Suchsystem zu ermöglichen, wird deshalb auf das Observer-Entwurfsmuster zurückgegriffen, das nach dem „publish-subscribe“-Verfahren funktioniert (vgl. Gamma et al.[19]). Das Suchsystem meldet sich beim Programmstart bei EUSOP als Beobachter an. Wenn veränderte Metadaten vorliegen, teilt EUSOP diese Änderung allen angemeldeten Beobachtern mit. Falls es keinen Beobachter gibt, also das Suchsystem nicht gestartet ist, so läuft diese Benachrichtigung ins Leere, was für EUSOP keine Auswirkungen hat. Die GUI des Suchsystems bezieht ihre Inhalte, also Benutzerinformationen und Suchergebnisse aus dem Fachkonzept des Suchsystems (4). Änderungen an Benutzerprofilen, Suchanfragen und Relevanzbewertungen werden von der Benutzeroberfläche an das Fachkonzept gesendet und dort verarbeitet (5). Wird ein Suchergebnis ausgewählt, wird die zur Verarbeitung der entsprechenden Aktion benötigte Methode im SiSeOr aufgerufen (6). Handelt es sich um eine Operation, wird diese in die Palette übernommen. Im Falle eines Dienstes werden die Metadaten aus dem EUSOP geladen (1) um die Eingabemaske damit zu füllen.

4.2 Datenhaltung

Das Suchsystem enthält verschiedene Informationen, die sich nicht ohne weiteres in den von EUSOP gebotenen Datenstrukturen unterbringen lassen. Aus diesem Grund verwaltet das System Daten wie Benutzerprofile, Relevanz-Bewertungen und in Schlagworte zerlegte Beschreibungstexte in Form von XML-Dateien selbst. XML-Dateien wurden hauptsächlich gewählt, da sich in ihnen die im Folgenden beschriebenen internen Datenstrukturen direkt ohne große Umformungen abbilden lassen und die Menschenlesbarkeit der Daten die Entwicklung erleichtert. Der folgende Abschnitt beschreibt, woher die vom Suchsystem verwendeten Informationen stammen, wozu sie dienen, welche programminterne Darstellung gewählt wurde und was persistent gespeichert werden muss.

Abbildung 4.5 zeigt ein Beispiel, das im Folgenden verwendet wird, um die vom Suchsystem angewendeten Verfahren zu erklären. Es handelt sich dabei um zwei Webservices, die Seekda.com[53] liefert, wenn nach Diensten zum Abrufen von Wetterberichten gesucht wird.

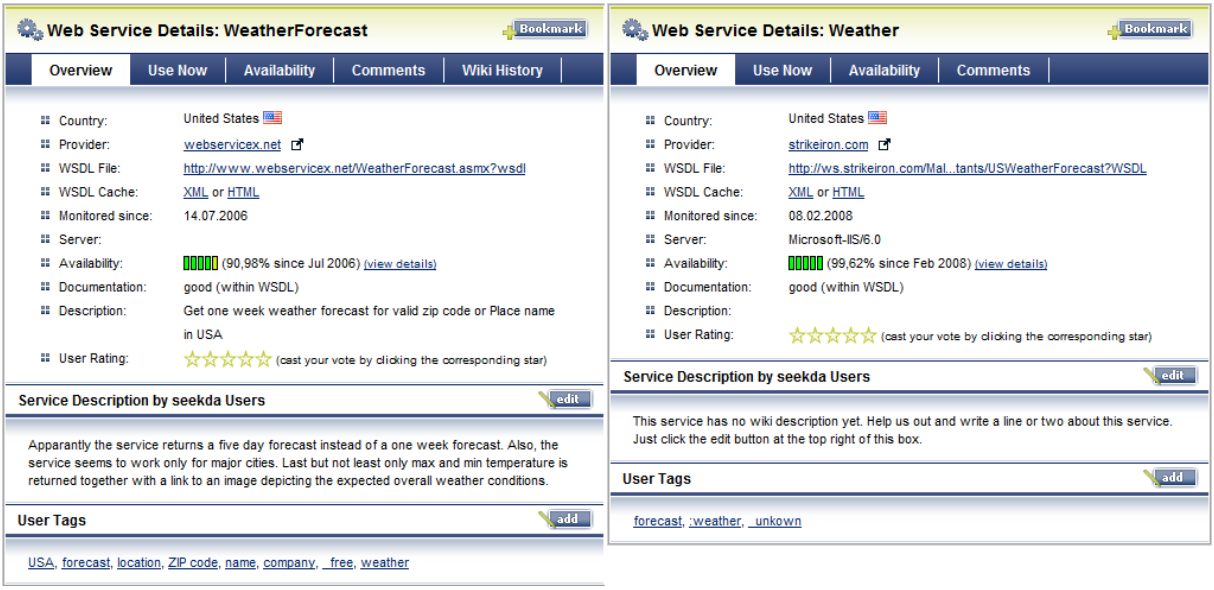

Abbildung 4.5: Beispiel: Dienstbeschreibungen auf Seekda.com[53]

Diese beiden Dienste wurden unter Verwendung der bei Seekda.com verwendeten Beschreibungstexte und Tags in EUSOP eingetragen, so dass beide Systeme die gleichen Informationen über die beiden Dienste enthalten und somit vergleichbar sind. Da Seekda.com sich allerdings wie auch andere, Web-basierte Service-Suchmaschinen auf die Dienst-Ebene konzentriert und keine benutzerdefinierten Operationsbeschreibungen enthält, wurden entsprechend keine Operationsbeschreibungen in EUSOP übernommen. Bei der Suche nach Testdaten hat sich allerdings gezeigt, dass Service-Anbieter in ihren WSDL-Dateien häufiger die einzelnen Operationen dokumentieren, als die Dienste selbst. Bei den vorhandenen Beschreibungen handelt es sich in der Regel um einzelne Sätze, die die Funktionsweise der Operation beschreiben. Häufig sind sie jedoch nicht genau genug um den Benutzer tatsächlich beim Verständnis des Dienstes zu unterstützen. Diese Beobachtung deckt sich daher mit der Annahme aus Kapitel 2.3, dass eine Benutzergemeinschaft langfristig Beschreibungen erzeugen kann, die für andere Benutzer hilfreicher sind, als Beschreibungen durch den Dienstanbieter. Wenn im WSDL bereits Beschreibungstexte enthalten sind, so werden diese von EUSOP automatisch übernommen und verwendet in der vorliegenden Form verwendet, solange sie nicht vom Benutzer verändert wurden. Auch die beiden hier gewählten Dienste bieten in den WSDL-Dateien Beschreibungen der Operationen, die dementsprechend zusätzlich zu den benutzerdefinierten Beschreibungen und Tags von Seekda.com verwendet werden. Da Seekda.com eine englischsprachige Seite ist werden im Folgenden auch in SiSeOr englische Beschreibungen verwendet.

4.2.1 Graphen

In Kapitel 3.2.1 wurden die Vorteile einer hierarchischen Darstellung der Dienst- und Operationsbeschreibungen erläutert. Um diese Graph-Metapher beizubehalten, wurden auch bei der Implementierung Graphen als Datenstruktur gewählt. Zur Umsetzung wurde JGraphT[28] gewählt, eine Open Source Bibliothek zur Darstellung von Graphen als Java-Objekte. JGraphT bietet Implementierungen gängiger Graphalgorithmen, die zur Berechnung der Ähnlichkeit benötigt werden. Der Vorteil von JGraphT im Vergleich zu anderen Java-Graph-Bibliotheken liegt vor allem in der guten Dokumentation und der Anbindung der JGraph-Bibliothek[27]. Diese ermöglicht eine einfache Visualisierung mit JGraphT erstellter Graphen, was insbesondere bei der Fehlersuche von Interesse ist und zur Erstellung der Abbildungen 4.6 und 4.7 genutzt wurde.

Das Suchsystem verwendet zwei Arten von Graphen, deren Inhalt und Aufbau den in Kapitel 3.2.1 vorgestellten und in den Abbildungen 3.7 und 3.8 gezeigten Graphen entspricht. Der erste gerichtete, ungewichtete Graph (Dienstgraph) enthält alle Dienste, Operationen und dazugehörige Schlüsselwörter. Er dient als Basis für die Berechnung der Ähnlichkeitswerte und als Suchmenge für die Schlüsselwortsuche (s. Kapitel 4.4.2). Die Aktualisierung des Dienstgraphen geschieht jedes mal, wenn eine Änderung an den Metadaten vorgenommen wird, da ein vollständiges Durchlaufen und Auslesen des gesamten UDDI-Verzeichnisses um einen vollständigen Graphen zu generieren nicht ohne weiteres möglich ist. Ein derartiger Vorgang würde viel Zeit in Anspruch nehmen und wie bereits in Kapitel 2.1.2 festgestellt wurde, sind nicht alle Datensätze im Verzeichnis auch tatsächlich funktionierende Dienst, was beim Auslesen berücksichtigt werden müsste. Abbildung 4.6 zeigt den Dienstgraphen für die beiden in Abbildung 4.5 vorgestellten Dienste. Die Abbildung zeigt die Dienste mit ihren Operationen, und die von ihnen aus erreichbaren Schlüsselwörter. Auf der linken Seite („Keywords Service 0“) befinden sich die Schlüsselwörter, die nur Teil der Beschreibung des Dienstes „WeatherForecast“ sind, auf der rechten Seite („Keywords Service 1“) sind alle Begriffe zu sehen, die nur den Dienst „Weather“ beschreiben. Die Schlüsselwörter in der Mitte von Abbildung 4.6 („Keywords Service0 & Service1“) sind in den Beschreibungen beider Dienste enthalten. Da beide Dienste über Operationen verfügen, die anhand von Postleitzahlen oder Städtenamen Wettervorhersagen abrufen können, handelt es sich dabei um Schlüsselwörter wie „weather“, „city“, „zip“ und „forecast“. Sämtliche Schlüsselwörter in der Abbildung wurden vor der Aufnahme in den Graphen von einem Stemming-Algorithmus bearbeitet, wodurch sie nicht in allen Fällen syntaktisch korrekten englischen Wörtern entsprechen.

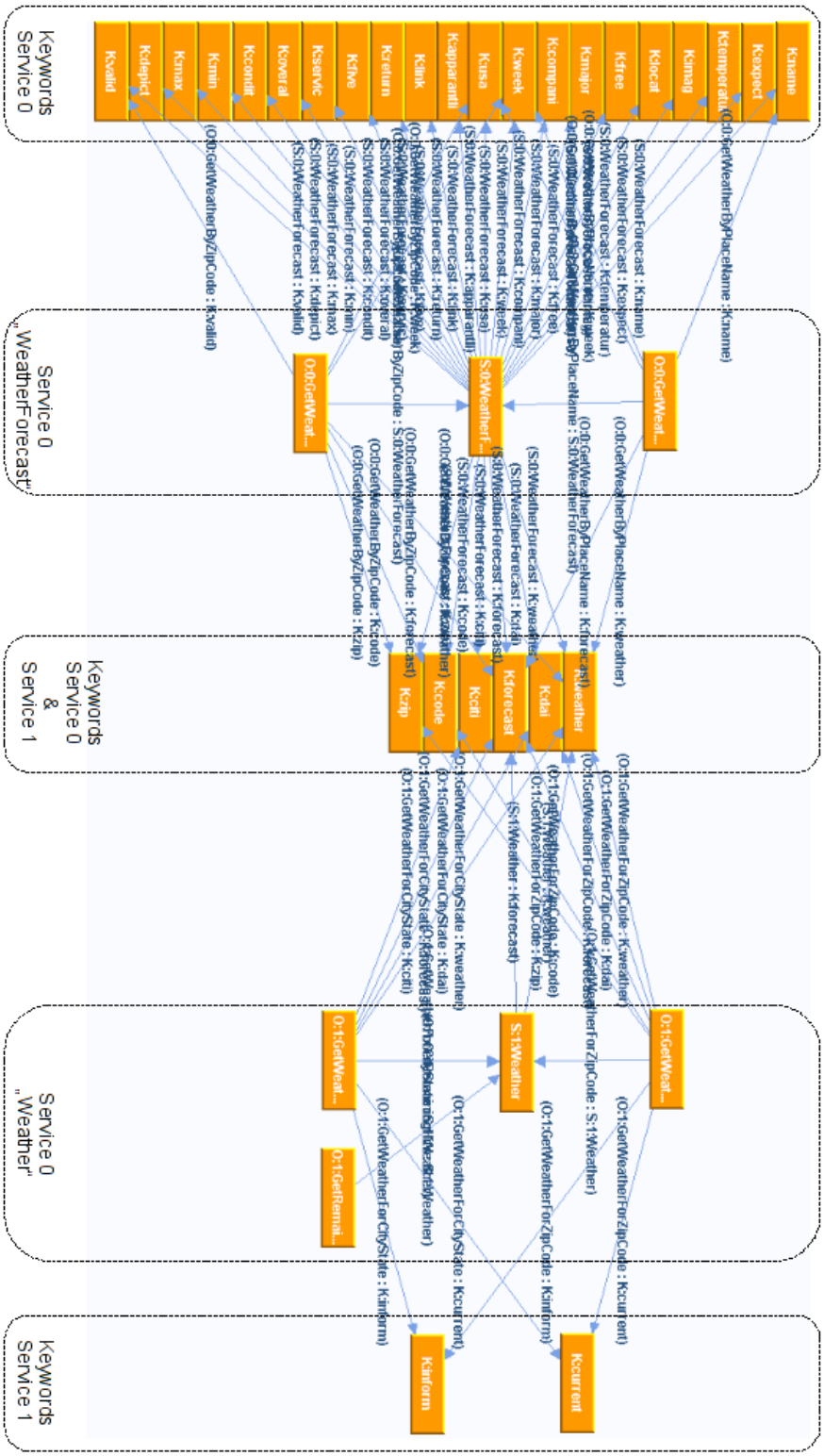

Abbildung 4.6: Beispiel: Dienstgraph mit zwei Diensten

Das Suchsystem muss zwischen Schlüsselwörtern und Diensten bzw. Operationen unterscheiden können. Aus diesem Grund enthalten die Knotennamen einen Präfix, der den Knotentypen bestimmt. „K" steht dabei für „Keyword", „S" für „Service" und „O" für „Operation". Des Weiteren gibt es noch „C" für „Category", Kategorien werden allerdings im Beispiel nicht genutzt, da ihre Eingabe und Auswahl von SiSeOr noch nicht unterstützt wird und auch Seekda.com, das als Quelle für die Beispieldaten diente keine Unterstützung für

Kategorien bietet. Die Zahl hinter dem Präfix dient zur eindeutigen Identifikation eines Dienstes und ermöglicht damit die Zuordnung von Operationen zu Diensten.

Der zweite, gerichtete und gewichtete Graph (Ähnlichkeitsgraph) enthält die Ähnlichkeiten zwischen Diensten bzw. Operationen. Da die Laufzeit der Berechnung der Ähnlichkeit zwischen sämtlichen Knoten des Dienstgraphen bei vielen Knoten sehr hoch werden kann, wäre es ineffizient diese bei jeder Suchanfrage durchzuführen. Bei der Berechnung wird für jeden Dienst und jede Operation eine Tiefensuche ausgeführt, um alle erreichbaren Knoten zu ermitteln. Die maximale Laufzeit einer einzelnen Tiefensuche liegt bei $O(|V|+|E|)$, wobei |V| für die Anzahl Knoten, und |E| für die Anzahl Kanten des Graphen steht. Diese Laufzeit kann in der Realität aufgrund der Eigenschaften der in Kapitel 3.2.1 vorgestellten Hierarchie des Dienstgraphen niemals erreicht werden, da von jedem Knoten aus nur eine relativ geringe Zahl anderer Knoten tatsächlich erreichbar ist. Dies liegt z.B. daran, dass von Diensten aus keine Operationen erreicht werden können und von den Schlüsselwortknoten, die einen großen Teil des Graphen ausmachen keine Suche ausgeht. Da allerdings der Graph sehr groß werden kann und aufgrund der schwer vorhersehbaren Zahl von verwendeten Schlüsselwörtern die tatsächliche Laufzeit nur schwer abgeschätzt werden kann, geschieht die Berechnung immer nur dann, wenn auch der Dienstgraph aktualisiert wird. Abbildung 4.7 zeigt den Ähnlichkeitsgraphen zum aktuellen Beispiel.

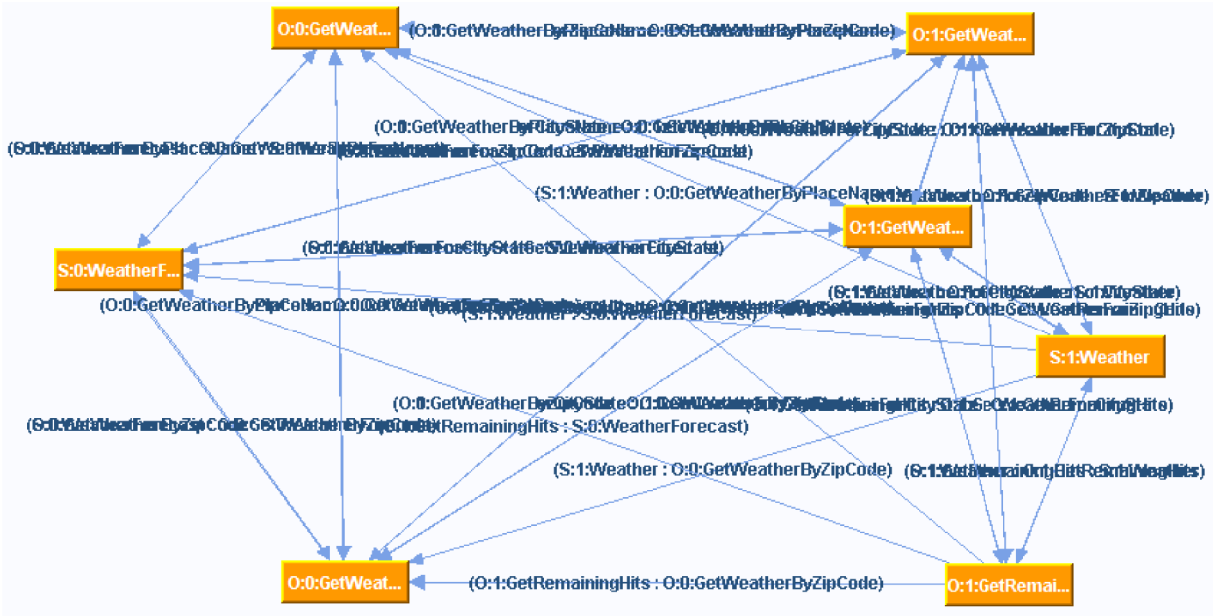

Abbildung 4.7: Beispiel: Ähnlichkeitsgraph

In der Abbildung sind alle Dienste und Operationen als Knoten dargestellt, die Kanten stehen für Ähnlichkeiten zwischen ihnen. Die Kantengewichte werden im Graphen nicht angezeigt.

Im Beispiel gibt es nur zwei Dienste, die über gemeinsam erreichbare Knoten verfügen.

Entsprechend lässt sich die Ähnlichkeit von jedem Dienst- bzw. Operationsknoten zu jedem anderen berechnen. Das Suchsystem bietet allerdings, wie in Kapitel 3.2.2 beschrieben, die Möglichkeit, eine minimale Ähnlichkeit festzulegen. Ergebnisse, die diesen Grenzwert unterschreiten, werden nicht als Suchergebnisse berücksichtigt und gespeichert. Dieser Fall tritt im vorliegenden Graphen beim Dienst „Weather“ und seiner Operation „GetRemainingHits“ auf. Die beiden Knoten verfügen selbst nur über wenige Schlüsselwörter, daraus folgt, dass Übereinstimmungen einzelner Schlüsselwörter einen größeren Einfluss auf die berechnete Ähnlichkeit haben, als bei ausführlicher beschriebenen Diensten. Dies führt im Beispiel dazu, dass einige Kanten der genannten Knoten nur als ausgehende Kanten vorliegen, weil in der Gegenrichtung die berechnete Ähnlichkeit zu niedrig ist.

Da eine vollständige Erfassung des UDDI-Verzeichnisses nicht möglich ist und die Berechnung sämtlicher Ähnlichkeitsbeziehungen innerhalb eines Graphen bei großen Datenmengen aufwändig werden kann, werden beide Graphen persistent gespeichert um unnötige Neuberechnungen zu vermeiden. Als Speicherformat wurde GraphML[23] gewählt, eine XML-basierte Sprache zur Darstellung von Graphen. Die Wahl fiel auf GraphML, da es im Gegensatz zu einigen vergleichbaren Formaten eine dem XML-Standard entsprechende Datei liefert und ein einfach zu verstehende Darstellung verwendet. GraphML-Dateien können von diversen Graph-Editoren (z.B. yEd[70]) gelesen werden. Des Weiteren wird das Format direkt von JGraphT unterstützt, allerdings können GraphML-Dateien nur geschrieben und nicht gelesen werden, und auch die Schreibfunktion lieferte nicht die gewünschten Ergebnisse, weshalb die entsprechenden Methoden neu implementiert werden mussten.

```
<graphml xsi:schemaLocation="http://graphml.graphdrawing.org/xmlns http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd">
  <graph id="ServiceGraph" edgedefault="directed">
    <node id="S:0:WeatherForecast">
      <desc>
        BCF4E4C0-BA32-11DD-A4C0-CD34CC75029F::Get one week weather forecast for valid zip code or Place name. Apparantly the service returns a
        five day forecast instead of a one week forecast. Also, the service seems to work only for major cities. Last but not least only max and min
        temperature is returned together with a link to an image depicting the expected overall weather conditions.
      </desc>
    </node>
    <node id="K:min"/>
    <node id="K:condit"/>
    <node id="K:valid"/>
      ...
    <node id="O:0:GetWeatherByZipCode">
      <desc>
        Get one week weather forecast for a valid Zip Code(USA)
      </desc>
    </node>
    <node id="S:1:Weather">
      <desc>5E62CCC0-BA31-11DD-8CC0-8FC29C992EDC::</desc>
    </node>
    <node id="O:1:GetRemainingHits"/>
    <node id="O:1:GetWeatherForZipCode">
      <desc>
        Gets current weather information and 2-day forecast for the given zip code
      </desc>
    </node>
    <node id="K:inform"/>
      ...
    <edge source="O:0:GetWeatherByZipCode" target="K:week"/>
    <edge source="S:1:Weather" target="K:weather"/>
    <edge source="O:1:GetRemainingHits" target="S:1:Weather"/>
    <edge source="O:1:GetWeatherForZipCode" target="S:1:Weather"/>
    <edge source="O:1:GetWeatherForZipCode" target="K:zip"/>
      ...
  </graph>
</graphml>
```

Abbildung 4.8: Beispiel: Auszug GraphML des Servicegraphen

Die Abbildungen 4.8 und 4.9 zeigen Auszüge aus den GraphML-Dateien der beiden Graphen. In GraphML-Dateien werden in „<node>“-Elementen die Knoten des Graphen definiert, und dann in „<edge>“-Elementen Kanten zwischen diesen Knoten dargestellt. Zusätzlich lassen sich die Elemente über das „<desc>“-Element dokumentieren, es erfüllt somit dieselbe Aufgabe wie das „<documentation>“-Element in WSDL-Dateien (vgl. Kapitel 2.1.1).

Dienst- und Operations-Knoten müssen eindeutig und voneinander unterscheidbar sein, da die jeweiligen Suchergebnisse unterschiedlich behandelt werden (vgl. Kapitel). Eine Operation kann immer nur zu genau einem Dienst gehören, und da nicht angenommen werden kann, dass Dienst- und Operationsnamen global eindeutig sind, enthalten ihre Knotennamen nach dem Typ-Präfix eine fortlaufende, bei Aufnahme in das Suchsystem automatisch vergebene Nummer, anhand derer zu jeder Operation eindeutig der Dienst bestimmt werden kann. Da innerhalb von Diensten Operationsnamen eindeutig sind ergibt sich aus Typ-Präfix, Servicenummer und Dienst-/Operations-Namen eine eindeutige Knotenbezeichnung. Falls Dienste und Operationen über Beschreibungstexte verfügen, werden diese in der GraphML-Datei des Servicegraphen (Abbildung 4.8) in den „<desc>“-Elementen abgelegt. Diese zusätzliche Speicherung der unverarbeiteten Beschreibungen dient der Beschleunigung der Suchfunktion. Teil der Suchergebnisse, die in der Benutzeroberfläche angezeigt werden, ist eine kurze Beschreibung des Suchergebnisses (vgl. Kapitel). EUSOP ruft bei jeder Anfrage nach Metadaten die WSDL des gewählten Dienstes ab. Da dieser Zugriff in der Regel über das Internet erfolgt, können sich daraus erhebliche negative Auswirkungen auf die Performance des Suchsystems ergeben. Bei normaler Nutzung der EUSOP-Plattform fällt dies kaum auf, da immer nur einzelne Beschreibungen angefragt werden. Wenn allerdings wie im vorliegenden Fall mehrere Beschreibungen nacheinander abgerufen werden, kann sich alleine durch die Übertragungszeit der WSDL-Dateien die Dauer einer Suchanfrage abhängig von der Verbindungsgeschwindigkeit im Extremfall auf mehrere Minuten verlängern. Die Speicherung der Beschreibungen in den GraphML-Dateien macht Zugriffe auf EUSOP während der Suche unnötig und geht damit diesem Problem aus dem Weg. Bei Dienst-Knoten enthalten die „<desc>“-Elemente außerdem noch den UDDI-Servicekey, der eine eindeutige Zuordnung von Suchergebnissen zu Einträgen im UDDI-Verzeichnis ermöglicht. Diese werden benötigt, um Suchergebnisse in der SiSeOr-Oberfläche weiterverwenden zu können.

Die GraphML-Datei des Ähnlichkeitsgraphen (Abbildung 4.9) entspricht im Aufbau dem des Dienstgraphen (Abbildung 4.8). Auf eine erneute Speicherung der Beschreibungstexte kann hier allerdings verzichtet werden. Da es sich beim Ähnlichkeitsgraphen um einen gewichteten Graphen handelt, wird zusätzlich in den „<edge>“-Elementen das Kantengewicht gespeichert. Zu diesem Zweck wird das „<data>“-Element verwendet, dass in

GraphML zur Erweiterung um beliebige, benutzerdefinierte Informationen genutzt wird.

```
<graphml xsi:schemaLocation="http://graphml.graphdrawing.org/xmlns http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd">
  <key id="d0" for="edge" attr.name="weight" attr.type="double"/>
  <graph id="SimilarityGraph" edgedefault="directed">
    <node id="S:0:WeatherForecast"/>
    <node id="O:0:GetWeatherByPlaceName"/>
    <node id="O:0:GetWeatherByZipCode"/>
      ...
    <edge source="S:0:WeatherForecast" target="O:0:GetWeatherByZipCode">
      <data key="d0">0.993</data>
    </edge>
    <edge source="S:0:WeatherForecast" target="O:1:GetWeatherForZipCode">
      <data key="d0">0.254</data>
    </edge>
      ...
```

Abbildung 4.9: Beispiel: Auszug GraphML des Ähnlichkeitsgraphen

4.3 Benutzerprofile und Relevanz-Feedback

Da es für Benutzer nicht zumutbar wäre, bei jeder Programmnutzung ihre Profile neu auszufüllen, werden auch diese in Form von XML-Dateien gespeichert. Ein Benutzerprofil besteht aus einem Benutzernamen, und einer Reihe von frei definierbaren Schlüssel-/Wert-Paaren. Grundsätzlich werden keine Felder vorgegeben, die verwendeten Schlüssel dienen jedoch dazu, Eigenschaften feste Namen zuzuweisen und so langfristig standardisierte, an eine Anwendungsdomäne angepasste Benutzerprofile zu ermöglichen. Dass die Felder nicht fest vorgegeben sind, führt zu einer sehr einfachen Struktur der verwendeten XML-Datei. Abbildung 4.10 zeigt eine Benutzerprofil-Datei mit einem einzelnen Profil, das über zwei Attribute verfügt und im Folgenden als Beispiel verwendet wird. Auf eine Anbindung der Profile an die in EUSOP bereits grundsätzlich vorhandene Benutzerverwaltung wird verzichtet, da sie vom SiSeOr-System bisher nicht verwendet wird. Allerdings lässt sich eine solche Anbindung nachträglich ohne weiteres einrichten, da die im Profil verwendeten Benutzernamen einen eindeutigen Identifikator darstellen, über den eine Verbindung zu möglicherweise irgendwann verfügbaren EUSOP-Benutzerkonten ermöglicht wird.

```
−<userprofiles xsi:schemaLocation="urn:nonstandard:userprofiles file:/./userprofiles.xsd">
  −<user id="Thorsten Theelen">
      <attribute id="Studiengang" value="Wirtschaftsinformatik"/>
      <attribute id="Beruf" value="Student"/>
    </user>
  </userprofiles>
```

Abbildung 4.10: Beispiel: XML-Datei mit einzelnem Benutzerprofil

Zur Erfassung der Relevanzbewertungen ist etwas mehr Aufwand nötig. Abbildung 4.11 zeigt eine zum Beispiel aus dem vorherigen Kapitel passende Feedback-Datei. Die

Suchergebnisse werden auch beim Feedback über ihren eindeutigen Knotennamen aus dem Servicegraphen identifiziert. Sobald eine Bewertung abgegeben wird, speichert das System zu jedem Suchbegriff, der zu diesem Ergebnis geführt hat, für jeden Begriff aus dem Benutzerprofil eine „Ja“- bzw. „Nein“-Stimme. Des Weiteren wird für die Suchbegriffe die Gesamt-Stimmenzahl erfasst, um die Berechnung der nicht personalisierten Relevanzvorhersage zu ermöglichen. Das genaue Verfahren zur Berechnung der Relevanz wird in Kapitel 3.2.3 erläutert. Im vorliegenden Beispiel wurde die Suchanfrage „weather forecast“ ausgeführt, während dem System die beiden Dienste aus dem vorherigen Abschnitt bekannt waren. Mit dem Benutzer aus Abbildung 4.10 wurde nun der Operation „GetWeatherForZipCode“ aus dem Dienst „Weather“ eine positive Stimme gegeben. Aus dieser Stimmabgabe ergibt sich die in Abbildung 4.11 gezeigte XML-Datei.

```
<feedback xsi:schemaLocation="urn:nonstandard:feedback file:/./feedback.xsd">
  <result id="O:1:GetWeatherForZipCode">
    <searchterm id="forecast" yes="1" no="0">
      <userterm id="Student" yes="1" no="0"/>
      <userterm id="Wirtschaftsinformatik" yes="1" no="0"/>
    </searchterm>
    <searchterm id="weather" yes="1" no="0">
      <userterm id="Student" yes="1" no="0"/>
      <userterm id="Wirtschaftsinformatik" yes="1" no="0"/>
    </searchterm>
  </result>
</feedback>
```

Abbildung 4.11: Beispiel: XML-Datei mit Benutzerfeedback

4.4 Komponenten

Um die Möglichkeit zu bieten, den Suchablauf nachträglich zu verändern, wurde bei der Entwicklung auf einen modularen Aufbau des Systems geachtet. Dies ermöglicht es, die verwendeten Algorithmen in Zukunft durch weitere Module zu ergänzen oder unter Umständen auch zu ersetzen, ohne das Gesamtsystem neu entwickeln zu müssen. Neben der Feedback-Komponente, deren Funktionsweise im wesentlichen im vorherigen Abschnitt, sowie in Kapitel 3.2.3 vorgestellt wurde, verfügt das System über die im Folgenden beschriebenen Module.

4.4.1 Textmining

Die Textmining-Komponente dient zur Verarbeitung der Community-generierten

Schlüsselwörter. Ihre Kernaufgaben sind dabei:

- Zerlegung von Texten in einzelne Schlüsselwörter
- Entfernung von Stoppwörtern
- Stemming

Als Basis zur Erfüllung dieser Aufgaben wurde WVTool[68] genutzt, eine frei verfügbare Java-Bibliothek, die verschiedene Funktionen zur Bearbeitung und statistischen Auswertung von Texten bietet. Für das vorliegende System sind dabei die in WVTool enthaltenen Klassen zur Stoppwortentfernung und zum Stemming von Bedeutung. Die Stoppwortentferung wurde letztendlich unter Verwendung des grundlegenden, in WVTool verwendeten Algorithmus neu implementiert, da die WVTool-Lösung eine fest programmierte Schlüsselwortmenge verwendet, während die neue Methode auf eine Textdatei mit Schlüsselwörtern zurückgreift, was eine Erweiterung und Anpassung der Stoppwortliste an domänen-spezifische Anforderungen ermöglicht. Das Textmining-Modul unterstützt grundsätzlich mehrere Sprachen. Im Falle der Stoppwörter wird dies durch die Nutzung einer anderen Stoppwortliste erreicht, für das Stemming verfügt WVTool über Algorithmen für verschiedene Sprachen, unter anderem den Porter-Stemming-Algorithmus(vgl. van Rijsbergen et al. [61]) für die englische Sprache, und ein auf dem in Kapitel 2.4.1 vorgestellten Verfahren von Caumanns[14] basierender Algorithmus für die deutsche Sprache.

4.4.2 Suchmodule

Das Suchsystem verfügt zur Zeit über zwei Suchmodule. Das erste führt eine Schlüsselwortsuche auf dem in Kapitel 4.2.1 vorgestellten Dienstgraphen durch. Sowohl die verwendeten Suchbegriffe, als auch die im Graphen enthaltenen Schlüsselwörter werden mit einem Stemming-Algorithmus auf ihre Grundform reduziert (vgl. Abbildung 4.6). Da bis auf seltene Ausnahmefälle verschiedene Formen eines Wortes auf die selbe Form reduziert werden (vgl. Caumanns [14]) sucht die Schlüsselwortsuche nach vollständigen Übereinstimmungen zwischen Suchbegriffen und Begriffen aus dem Dienstgraphen, wobei die Punktzahl nach der in Kapitel 3.2.2 vorgestellten Formel berechnet wird.

Das zweite Suchmodul, die Ähnlichkeitssuche, verwendet die Suchergebnisse der Schlüsselwortsuche und liefert alle Dienste oder Operationen, die eine berechenbare Ähnlichkeit zu den direkten Treffern haben und über dem festgelegten Grenzwert liegen. Die zur programminternen Darstellung der Graphen verwendete JgraphT-Bibliothek[28] bietet

eine Methode zur Durchführung einer Tiefensuche, die es ermöglicht, alle von einem Knoten aus in Kantenrichtung erreichbaren Knoten zu ermitteln. Diese Information wird genutzt, um nach dem in Kapitel 3.2.2 vorgestellten Verfahren die Ähnlichkeit der Knoten im Graphen zu errechnen.

4.4.3 Strategien und Konfiguration

Um Veränderung des Suchablaufs zu erleichtern, nutzt das Suchsystem das Strategy-Entwurfsmuster (vgl. Gamma et al.[19]). Dieses bietet eine einheitliche Schnittstelle für mehrere mögliche Algorithmen zur Lösung eines Problems. Im vorliegenden Fall wird für Suchstrategien lediglich eine Schnittstelle vorgegeben, die eine Menge von Suchbegriffen erhält, und eine sortierte Liste von Suchergebnissen zurückliefert. Konkrete Suchstrategien können aus Kombinationen vorhandener Modulen, oder auch aus neuen Suchmodulen bestehen. Die in der vorliegenden Version des Suchsystems verwendete Suchstrategie führt wie bereits beschrieben zuerst eine Schlüsselwortsuche durch, dann eine Ähnlichkeitssuche und sortiert die Ergebnisse anschließend mit Hilfe des Feedback-Systems (vgl. Abbildung). Zur Auswahl der Strategie wird eine zentrale Klasse verwendet, die alle für den Programmablauf relevanten Parameter enthält. Darunter befindet sich auch die verwendete Sprache des Textmining-Moduls, sowie alle in Kapitel 3.2.2 vorgestellten Parameter zum Einstellen der Such- und Ranking-Algorithmen.

4.5 Benutzeroberfläche

Das Suchsystem verwendet zwei eigene Eclipse-Views, die als Registerkarten in dasselbe Unterfenster der SiSeOr-Oberfläche eingebunden werden wie die bisherige Suchfunktion. Da SiSeOr, wie bereits besprochen, bislang über keine eigene Benutzerverwaltung verfügt wurde als erstes eine Eingabemaske zur Auswahl des Benutzers erstellt. Der in Abbildung 4.12 dargestellte Bildschirm ermöglicht die Auswahl eines Benutzers, das Erstellen eines neuen Benutzers sowie das Hinzufügen und Ändern von Benutzerbegriffen. Da die Benutzerprofile bislang an keine Zugriffsrechte gebunden sind und die Benutzerverwaltung letztendlich ein untergeordnetes Thema dieser Arbeit ist, wurde auf die Verwendung weiterer Funktionen zur Benutzerverwaltung, sowie von Passwörtern und anderen Sicherheitsmechanismen verzichtet.

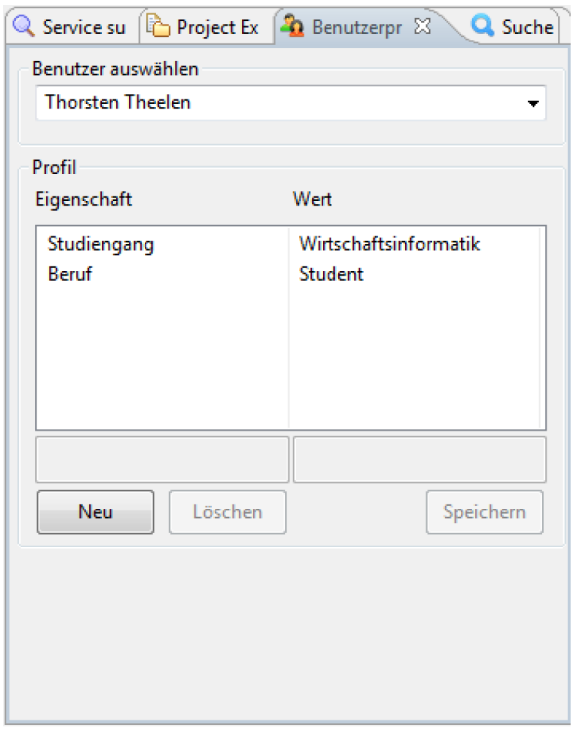

Abbildung 4.12: Benutzerverwaltung

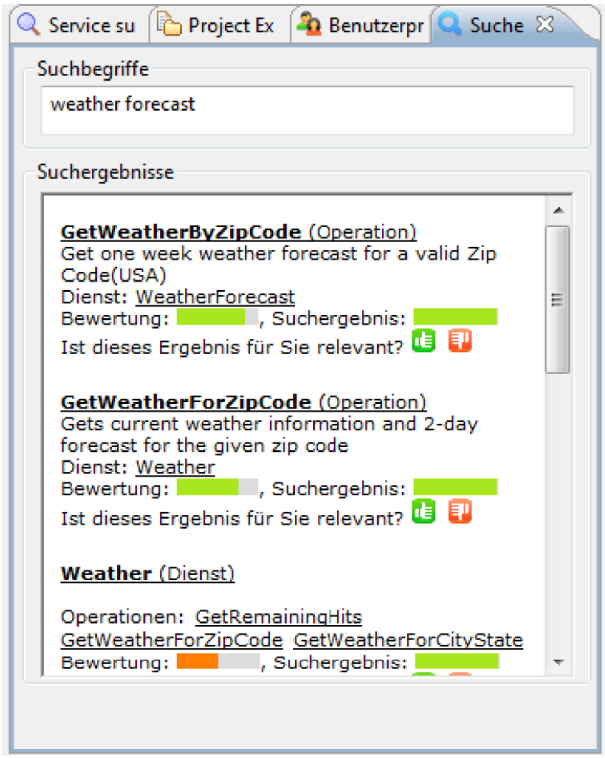

Abbildung 4.13: Suchergebnisse

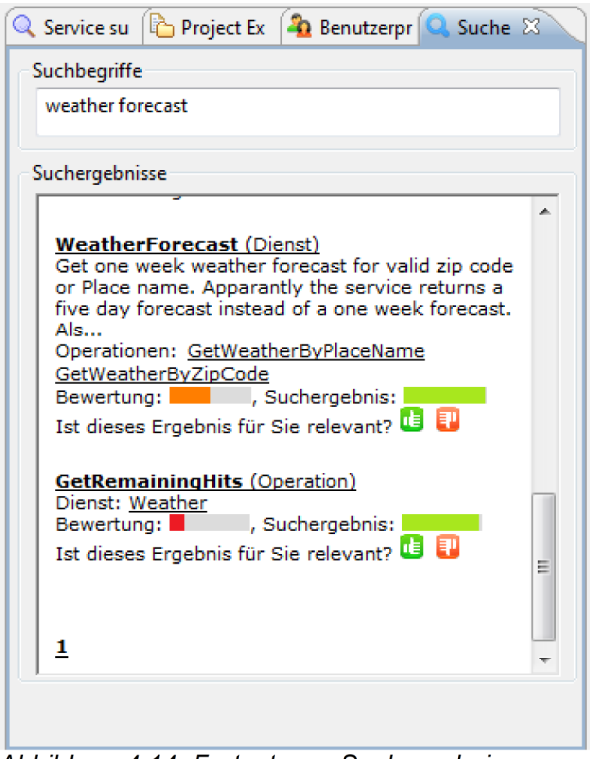

Abbildung 4.14: Fortsetzung Suchergebnisse

Der Schwerpunkt bei der Entwicklung der Benutzeroberfläche lag bei dem in den Abbildungen 4.13 und 4.14 gezeigten Suchbildschirm. Die Hauptanforderung an die Benutzeroberfläche war die Orientierung am Aussehen und der Bedienung von webbasierten Suchmaschinen. Die Eingabe der Suchbegriffe erfolgt wie gefordert über ein einfaches Textfeld, in das beliebige Suchbegriffe eingegeben werden können. Es gibt zwei Arten von Suchergebnissen, Dienste und Operationen. Ein Dienstergebnis enthält den Namen des Dienstes, den Beschreibungstext sowie eine Liste der enthaltenen Operationen. Ein Operationsergebnis besteht aus dem Namen der Operation, der Operationsbeschreibung und dem Dienst zu dem die Operation gehört. Zusätzlich verfügen beide Ergebnistypen über eine Zeile, die in Form von farbigen Balken die Qualität des Suchergebnisses beschreibt. Der linke Balken steht dabei für die für die personalisierte Relevanzvorhersage, die sich aus den Relevanzbewertungen der Benutzercommunity errechnet. Der rechte Balken steht für die von den Suchalgorithmen berechnete Punktzahl. Die letzte Zeile von jedem Ergebnis bietet den Benutzern die Möglichkeit eigene Relevanzbewertungen abzugeben. Bewegt der Benutzer den Mauszeiger über ein interaktives Element in der Ergebnisliste, so wird in einem Tooltip eine Beschreibung der ausführbaren Aktion angezeigt. Die zu den Ergebnisbalken gehörende Beschreibung enthält die genauen Punktzahlen.

Sämtliche Benutzeraktionen werden, wie bei Webseiten üblich, durch einen Einzelklick ausgeführt. Ein Klick auf einen Dienstnamen zeigt die Metadaten des Dienstes in der SiSeOr-Oberfläche an. Wird eine Operation angeklickt, so wird diese in die SiSeOr-Palette

zur Weiterverarbeitung im Orchestrierungseditor übernommen. Zur Verbesserung der Übersichtlichkeit lässt sich das Suchergebnis, wie auch bei Web-Suchmaschinen üblich in mehrere Seiten aufteilen, um die Anzeige zu vieler Suchergebnisse auf einem Bildschirm zu vermeiden.

Die Umsetzung einer Ergebnis-Liste, die sich an dem Aufbau von Web-Suchmaschinen orientiert, erwies sich bei ausschließlicher Verwendung von SWT-Elementen als schwierig. SWT bietet Textfelder, Tabellen und Baumstrukturen als mögliche Elemente zum Anzeigen von Suchergebnissen. Diese ermöglichen es jedoch kaum mehrzeilige Ergebnisse zu verwenden, innerhalb einer Zeile unterschiedliche Formatierungen zu verwenden und verschiedenen Elementen einer Zeile unterschiedliche Aktionen zuzuordnen. Um das gesetzte Ziel trotzdem zu erreichen, wurde eine auf HTML-basierende Darstellung der Ergebnisse gewählt, die zwar die direkte Kommunikation mit der Java-Benutzeroberfläche erschwert, allerdings die notwendige Flexibilität bietet. Nach Durchführung der Suche generiert das Programm aus den Ergebnissen HTML-Code, der dann an ein SWT-Element zur Anzeige von HTML-Seiten übergeben wird. Dieser SWT-Browser bietet grundsätzlich alle Funktionen, die zur Realisierung eines vollwertigen Web-Browsers benötigt werden. Im vorliegenden Fall wird zwar nur die reine HTML-Darstellung benötigt, allerdings wird die Möglichkeit, von Webseiten aus Statusmeldungen an den Web-Browser zu schicken, zur Kommunikation mit der Java-Oberfläche verwendet. Mit Funktionen belegte Elemente der HTML-Seite senden über eine Javascript-Funktion eine Statusmeldung an den Browser. Diese wird von der Java-Oberfläche abgefangen und ausgewertet, um die angeforderten Operationen auszuführen.

5 Evaluation

Zur Evaluation des entwickelten Suchsystems wurde ein Benutzungstest durchgeführt, der im Folgenden vorgestellt wird. Zuerst werden die Ziele der Evaluation beschrieben und die Methodik des Benutzungstests vorgestellt. Danach werden die Ergebnisse ausgewertet und es wird auf Beschränkungen des durchgeführten Tests hingewiesen.

5.1 Ziele und Methodik

Ziel des Benutzungstests war es, die Funktionalität und Benutzerfreundlichkeit des Suchsystems im Vergleich zu einer webbasierten Webservice-Suchmaschine zu betrachten, um herauszufinden, mit welchen Aspekten des Suchsystems die Benutzer zufrieden sind und wo noch Verbesserungen möglich wären. Web-Suchmaschinen bieten sich als Vergleichssysteme an, da sie, wie bereits festgestellt wurde, zur Zeit die erfolgversprechendste Methode zur Suche nach Webservices darstellen und außerdem bei der Entwicklung des Suchsystems versucht wurde, das Verhalten von Web-Suchmaschinen nachzubilden.

Die grundsätzliche Methodik wurde dabei aus der Arbeit „A comprehensive and systematic model of user evaluation of web search engines“ von L.T. Su[57][58] übernommen. In dieser Arbeit wird ein Verfahren vorgestellt, um Web-Suchmaschinen durch Testpersonen miteinander vergleichen zu lassen. Die Teilnehmer führen dazu auf den verschiedenen Suchmaschinen ihre alltäglichen Suchanfragen durch und bewerten anhand von Fragebögen die Qualität der Ergebnisse sowie die Performance und Benutzerfreundlichkeit der verglichenen Systeme. Dabei kommen zwei Arten von Fragebögen zum Einsatz. Der erste Fragebogen wird während der eigentlichen Suche pro Suchmaschine ausgefüllt und beschäftigt sich mit der Bedienung der Suchmaschine, sowie der Geschwindigkeit, in der Suchanfragen bearbeitet werden. Der zweite Fragebogen wird ausgefüllt, nachdem alle Suchmaschinen verwendet wurden. In ihm werden die verschiedenen Systeme miteinander verglichen. Im einzelnen wird behandelt, wie zufrieden die Teilnehmer mit der Relevanz der gelieferten Suchergebnisse sind und wie gut die Benutzer ihre Ziele mit Hilfe der verschiedenen Suchmaschinen erreichen konnten.

5.2 Setting und Ablauf

Als Teilnehmer des Tests standen acht Wirtschaftsinformatik-Studenten zur Verfügung, die als Gegenleistung ein Mittagessen erhielten. Der Test wurde an zwei Nachmittagen durchgeführt und dauerte jeweils 1-2 Stunden, abhängig von der Anzahl der Teilnehmer an den beiden Terminen. Der Benutzungstest wurde in einem Rechnerraum der Universität Siegen durchgeführt. Es standen drei PC's zur Verfügung, auf denen SiSeOr mit dem neuen Suchsystems installiert war, sowie weitere Computer mit Internetzugang, die zur Verwendung von Seekda.com verwendet wurden. Als Betriebssystem wurde auf allen Rechnern Windows XP verwendet.

Der Ablauf des Tests, sowie die Fragebögen wurden im wesentlichen aus der vorgestellten Arbeit übernommen. Da jedoch keine allgemeinen Web-Suchmaschinen, mit deren Verwendung die meisten Benutzer vertraut sind, sondern Webservice-Suchmaschinen, mit denen im alltäglichen Leben nur wenige Menschen in Kontakt kommen, verglichen wurden, mussten folgende Anpassungen vorgenommen werden. Da davon ausgegangen wurde, dass nur wenige Teilnehmer des Benutzungstests regelmäßig nach Webservices suchen wurde das folgende Suchszenario vorgegeben:

„Gesucht ist ein Webservice, der die Möglichkeit bietet anhand eines beliebigen, internationalen Ortsnamen den Wetterbericht für diesen Ort für die nächsten Tage zu liefern.“

Auch an den Fragebögen wurden geringfügige Änderungen vorgenommen. So wurde in der Originalstudie mit Ausdrucken der Ergebnislisten gearbeitet. Dieses Vorgehen wäre im Rahmen des Benutzungstests schwierig gewesen, und da der gesamte Test innerhalb eines kurzen Zeitraums stattfand und nur zwei Systeme verglichen wurden, wurde darauf verzichtet. Des Weiteren wurde in der Originalstudie anhand von Stichproben getestet, ob die als Ergebnisse gelieferten Webseiten überhaupt funktionieren. Ein Webservice kann im Gegensatz zu Webseiten nicht per Klick auf ein Suchergebnis direkt ausgeführt werden. Um die Funktionsfähigkeit bzw. Verfügbarkeit des Dienstes zu überprüfen würde eine spezielle Testumgebung benötigt, deshalb wurde dieser Schritt im Rahmen des Benutzungstests nicht ausgeführt. Die Fragebögen liegen in der ursprünglichen Arbeit in englischer Sprache vor. Auf eine Übersetzung wurde verzichtet, so dass auch im Benutzungstest mit englischen Fragebögen gearbeitet wurde.

Als Vergleichssystem wurde Seekda.com[53] gewählt, ein Webservice-Verzeichnis, das verschiedene Suchfunktionen anbietet. Seekda.com bietet der Benutzergemeinschaft die Möglichkeit, die Dienste zu beschreiben. Dazu dient wie in Abbildung 5.1 erkennbar ist, ein

Wiki-System mit Dienstbeschreibungen (1), sowie die Möglichkeit den Diensten Tags zuzuweisen (2).

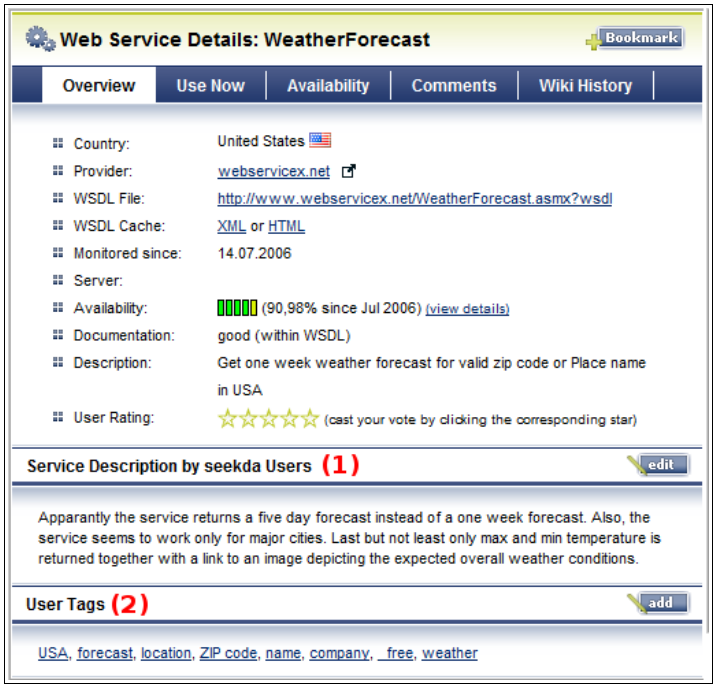

Abbildung 5.1: Dienstbeschreibung auf Seekda.com[53]

Die zur Verfügung stehenden Informationen entsprechen also den vom neuen Suchsystem verwendeten Daten. Um das Suchsystem mit Testdaten zu füllen, wurden eine Reihe von Suchanfragen auf Seekda.com ausgeführt, die zum Suchszenario passende Ergebnisse liefern. Die Suchergebnisse von Seekda.com wurden dann in das SiSeOr-basierte Suchsystem eingetragen, inklusive der benutzergenerierten Beschreibungstexte und Tags. Zusätzlich wurden noch eine Reihe von Diensten eingetragen, die keinen direkten Bezug zum Wetter haben, aber Ähnlichkeiten in der Beschreibung aufweisen, wie z.B. Dienste zur Suche nach Postleitzahlen. Die von diesen Diensten gebotenen Operationen verwenden häufig Postleitzahlen und Ortsnamen, könnten daher abhängig von der verwendeten Suchanfrage ebenfalls als Ergebnis geliefert werden. Insgesamt enthielt das neue Suchsystem ca. 50 Webservices mit insgesamt ca. 450 Operationen. Seekda.com liefert wie die meisten Webservice-Suchsysteme nur Dienste als Ergebnisse, entsprechend können auf der Webseite auch nur die Dienste selbst beschrieben werden und nicht die Operationen. WSDL-Dateien enthalten jedoch, wie bereits in einem früheren Kapitel festgestellt wurde, oft kurze Operationsbeschreibungen durch die Dienstanbieter. Diese Beschreibungen werden, solange es keine Beschreibungen durch Benutzer gibt beim Einlesen der WSDL-Dateien durch SiSeOr automatisch übernommen, was dazu führte, dass das Suchsystem zusätzlich zu den von Seekda.com übernommenen Beschreibungen über Operationsbeschreibungen verfügte.

Zu Beginn des Benutzungstests wurden die Fragebögen ausgeteilt und das Suchszenario erklärt. Die Teilnehmer verwendeten dann nacheinander die beiden Suchsysteme und füllten dabei jeweils einen Fragebogen aus. Um einen Einfluss von Lerneffekten durch die aufeinanderfolgende Verwendung der beiden Systeme auf das Ergebnis des Tests zu verhindern, nutzte die Hälfte der Teilnehmer zuerst Seekda.com, die andere Hälfte SiSeOr. Während der Verwendung der Suchsysteme sollten die Benutzer 3-5 zum Suchszenario passende Suchstrategien auswählen und ausführen. Dabei konnte es sich um verschiedene Kombinationen aus Suchbegriffen handeln, die Verwendung der erweiterten Suchfunktionen von Seekda.com war ebenfalls zugelassen. Die Teilnehmer wählten daraufhin die Suchstrategie, die ihrer Meinung nach insgesamt die besten Ergebnisse lieferte und zählten die relevanten, teilweise relevanten und nicht relevanten Ergebnisse. Daraufhin beantworteten sie die übrigen Fragen zur Performance und Bedienung des jeweiligen Suchsystems. Nach der Durchführung dieser Schritte für beide Systeme füllten die Teilnehmer einen weiteren Fragebogen aus, in dem die beiden Suchsysteme miteinander verglichen wurden und insbesondere die Zufriedenheit der Benutzer mit der Relevanz der Suchergebnisse und die Unterstützung bei der Erfüllung ihres Suchziels durch die Systeme beurteilt wurden.

5.3 Qualitative Auswertung der Fragebögen

Die Fragebögen, die vollständig im Anhang dieser Arbeit zu finden sind, enthielten zu jeder Frage eine 7-Punkte-Skala zur Bewertung der einzelnen Aspekte, sowie ein Feld, in das die Teilnehmer Bemerkungen zu den jeder Fragestellung eintragen konnten. Aufgrund der geringen Teilnehmerzahl schien eine statistische Auswertung der vergebenen Punkte als nicht sinnvoll, stattdessen wurden hauptsächlich die Bemerkungen der Benutzer betrachtet. Da die Fragebögen weitgehend aus der bereits vorgestellten Arbeit von L.T. Su[57] übernommen wurden, lagen sie in englischer Sprache vor. Dies führte vermutlich in Kombination mit der Tatsache, dass die Teilnehmer des Benutzungstests nicht viel Erfahrung mit Webservices hatten dazu, dass die Antworten nicht immer völlig zur Fragestellung passten. Deshalb wurden die Antworten der Benutzer unabhängig von den Fragen, auf die sie ursprünglich gegeben wurden, in Kategorien eingeteilt. Der Benutzungstest sollte das System in Hinblick auf Benutzerfreundlichkeit und Funktionalität beurteilen, entsprechend wurden die Bemerkungen der Nutzer anhand dieser beiden Kriterien zusammengefasst. Des Weiteren lies sich häufig beobachten, dass die Teilnehmer bei beiden Suchmaschinen dieselben Aspekte hervorhoben. Wenn also bei einem System eine bestimmte Eigenschaft als positiv oder negativ aufgefallen ist, wurde häufig bei der Beurteilung des zweiten Suchsystems dieselbe Eigenschaft hervorgehoben, so entsprach

z.B. oft eine negative Bewertung des einen Systems einer positiven Bewertung des anderen. Auf diese Weise fließen die Bemerkungen zu Seekda.com in die Auswertung der Fragebögen zum SiSeOr-basierten Suchsystem ein, sie werden hier jedoch nicht weiter explizit behandelt, da es im Benutzungstest nicht um eine Beurteilung von Seekda.com ging. Im Folgenden werden die wesentlichen Kritikpunkte am Suchsystem zusammengefasst.

Funktionalität:

- *niedrige Präzision*: Der Anteil an nicht relevanten Suchergebnissen war aus Sicht einiger Benutzer zu hoch. Einigen Teilnehmern erschien es so, dass die Suchanfrage vom System nicht richtig verstanden wurde und die Suche eher nach passenden Diensten eher einem „Trial and Error"-Vorgehen entsprach. Dies wurde bei beiden Suchsystemen bemängelt.

- *Sortierung*: Die Sortierung der Suchergebnisse entsprach nicht immer der Einschätzung der Relevanz durch die einzelnen Teilnehmer. Des Weiteren wurde durch den Benutzer veränderbare Sortierung nach verschiedenen Kriterien gewünscht.

- *Keine erweiterten Suchfunktionen*: Seekda.com verfügt über eine „Advanced Search"-Funktion, in der explizit nach bestimmten Eigenschaften der Dienste gesucht werden kann. Auch wenn kein Teilnehmer die Verwendung dieser erweiterten Suche als beste Suchstrategie empfunden hat, erschien sie mehreren Benutzern als sinnvolle Erweiterung für das Suchsystem. Des Weiteren entspricht die Suche beim neuen Suchsystem einer „oder"-Suche, d.h. es werden alle Dienste bzw. Operationen als Ergebnisse geliefert, die mindestens einen der gesuchten Begriffe enthalten oder einem anderen Ergebnis sehr ähnlich sind, die mindestens einen der Begriffe enthalten. Das bedeutet, dass mit zusätzlichen Suchbegriffen die Anzahl der Suchergebnisse ansteigt, während die Erwartungshaltung einzelner Teilnehmer war, dass durch zusätzliche Suchbegriffe Ergebnisse herausgefiltert werden müssten, die diesen Begriff nicht enthalten.

Benutzerfreundlichkeit:

- *Keine Onlinehilfe:* Das Suchsystem verfügt abgesehen von Tooltips, die in der Ergebnisliste die interaktiven Elemente erläutern, über keine Onlinehilfe. Die Fragebögen enthielten eine Frage, in der die Qualität der Onlinehilfe bewertet werden sollte. Da es keine echte Hilfe gab, fielen die Bewertungen entsprechend schlecht auf, wobei von einzelnen Benutzern angemerkt wurde, dass aufgrund der einfachen Bedienung der Suchfunktion keine Onlinehilfe notwendig wäre.

- *Unterscheidung zwischen Diensten und Operationen:* Nur wenigen Benutzern ist aufgefallen, dass es zwei verschiedene Arten von Ergebnissen gab. Dies kann zum einen daran liegen, dass der Unterschied zwischen Diensten und Operationen aufgrund der fehlenden Erfahrung der Teilnehmer mit Webservices nicht allen klar war, es wurde jedoch zum Teil auch direkt kritisiert, dass die Unterscheidung zwischen den beiden Ergebnisarten in der Ergebnisliste zu schwer fällt, so dass es möglicherweise einfach von einigen übersehen wurde.

- *SiSeOr-Anbindung/-Layout*: Einige Kritikpunkte bezogen sich mehr auf SiSeOr als ganzes, als auf das Suchsystem selbst. So wurde mehrfach die Größe und Anordnung des Suchfensters innerhalb von SiSeOr kritisiert. Des Weiteren war die Anbindung der Ergebnisliste an die SiSeOr-Servicedetails nicht für jeden erkennbar, so dass mehrere Benutzer der Meinung waren, dass bei einem Klick auf ein Suchergebnis keine Aktion ausgeführt würde. Auch dies lässt sich möglicherweise mit der fehlenden Erfahrung mit Webservices und der SiSeOr-Umgebung erklären, sowie der Erwartungshaltung von Web-Suchmaschinen, dass sich eine neue Seite öffnet, wenn ein Link angeklickt wird und nicht nur eine nicht direkt sichtbare Aktion ausgeführt wird.

- *Weitere Kritikpunkte an der Benutzeroberfläche:* Es gab noch ein paar weitere kleine Verbesserungsvorschläge für die Benutzeroberfläche, so haben z.B. mehrere Teilnehmer das Fehlen eines Knopfes zum Abschicken der Suchanfrage bemängelt. Zur Zeit werden Suchanfragen mit der „Enter“-Taste abgeschickt. Zwar hatte während des Benutzungstests niemand Probleme mit dieser Art der Bedienung, gerade unerfahrene Computernutzer könnten jedoch Schwierigkeiten mit diesem Vorgehen haben. Ein weiterer Kritikpunkt von einzelnen Teilnehmern war, dass die Gesamtzahl der gefundenen Suchergebnisse in der Ergebnisliste nicht angezeigt wird.

Es gab nicht nur Kritik und Verbesserungsvorschläge zum Suchsystem, einige Aspekte wurden auch positiv hervorgehoben:

- *Geschwindigkeit:* Aufgrund der separaten Datenhaltung des Suchsystems und der Berechnung der Ähnlichkeiten bei Veränderungen an den Dienstbeschreibungen sind die Suchanfragen sehr schnell, was von den Benutzern positiv bewertet wurde.

- *Bedienung*: Die Bedienung des Suchsystems wurde als sehr einfach empfunden. Kritikpunkte an der eigentlichen Bedienung standen fast immer im Zusammenhang mit der Anbindung an SiSeOr (s.o.).

- *Unterscheidung Dienste/Operationen*: Dies wurde, wie bereits beschrieben, nur von wenigen Benutzern überhaupt wahrgenommen. Diejenigen, die es bemerkt haben,

hielten es jedoch für eine sinnvolle Verbesserung im Vergleich zu Seekda.com, wo nur Dienste berücksichtigt werden.

- *Servicedetails in Ergebnisliste/SiSeOr:* Ein Kritikpunkt an Seekda.com war, dass in der Ergebnisliste selbst keine Dienstbeschreibungen angezeigt werden, sondern nur der Titel des Dienstes. Um die Beschreibung zu lesen, muss die Detailseite des Dienstes geöffnet werden. Entsprechend ist die Tatsache, dass beim SiSeOr-Suchsystem direkt in der Ergebnisliste kurze Beschreibungen angezeigt werden, positiv beurteilt worden. Des Weiteren wurden auch die in SiSeOr angezeigten, erweiterten Servicedetails als positiv empfunden. Eine weitere positive Anmerkung von mehreren Benutzern, dass im Gegensatz zu Seekda.com auch wirklich alle Dienste über eine Beschreibung verfügten, lässt sich einfach mit der Tatsache begründen, dass Dienste ohne Beschreibung aufgrund der Funktionsweise des Suchsystems nicht als Ergebnisse geliefert werden können.

Neben der erweiterten Suchfunktion, verwendeten mehrere Benutzer die zusätzlich von Seekda.com angebotenen Funktionen für Tags. So besteht z.B. die Möglichkeit auf den Detailseiten der Dienste die verwendeten Tags direkt anzuklicken um so eine Liste von anderen Diensten zu erhalten, die ebenfalls mit dem angeklickten Schlagwort beschrieben werden. Das Fehlen dieser Funktionen wurde zwar nicht explizit am neuen Suchsystem kritisiert, es wurde jedoch von einzelnen Teilnehmern als die beste, bei Seekda.com zur Verfügung stehende Suchstrategie empfunden und könnte entsprechend auch als mögliche Verbesserung für das SiSeOr-Suchsystem betrachtet werden.

5.4 Beschränkungen

Da dieser Benutzungstest Teil einer Diplomarbeit war und dementsprechend die zur Verfügung stehende Zeit und Teilnehmerzahl beschränkt war, konnten einige Aspekte des Suchsystems nicht getestet werden. Eine Grundidee der Arbeit ist, dass sich die Vollständigkeit und Korrektheit der verwendeten Community-generierten Inhalte langfristig ergibt, wenn genügend Mitglieder der Benutzergemeinschaft Inhalte beisteuern. Für den Benutzungstest wurden die Community-generierten Beschreibungen und Tags von Seekda.com übernommen und außerdem die Operationsbeschreibungen der Anbieter aus den WSDL-Dateien verwendet. Über die Qualität der verwendeten Datenbasis lässt sich daher kaum eine Aussage treffen.

Ein weiterer Aspekt des Suchsystems ist das Relevanz-Feedback, dass dafür sorgen soll, dass sich die Sortierreihenfolge der Suchergebnisse langfristig der Einschätzung der Benutzer annähert und somit Ergebnisse, die nicht relevant sind, an das Ende der Ergebnisliste wandern und ggf. völlig herausgefiltert werden können. Vor dem Benutzungstest wurden sämtliche Relevanzbewertungen entfernt, so dass die Sortierung vollständig vom Suchalgorithmus übernommen wurde, wodurch eine geringe Präzision abzusehen war (vgl. Kapitel 3.2.2). Die Verwendung vom Relevanzfeedback wäre außerdem eine Voraussetzung für die Überprüfung der Effekte der Personalisierung gewesen, die entsprechend auch nicht durchgeführt werde. Des Weiteren hätte dazu jeder Benutzer ein eigenes Profil benötigt, und bei nur acht Benutzern wären die Effekte durch ähnliche Profile vermutlich gering gewesen. Zur Vereinfachung benutzten daher sämtliche Teilnehmer dasselbe Benutzerprofil, eine Personalisierung fand demnach nicht statt.

Während des Benutzungstests stellte sich heraus, dass es sehr schwer ist, die Relevanz eines Webservice zu beurteilen. Da sich die Dienste, wie bereits beschrieben, nicht ohne eine Art von Testumgebung direkt vom Benutzer ausprobieren lassen, blieb zur Beurteilung nur die vorhandene Beschreibung. Entsprechen schwankten die subjektiven Einschätzungen der Teilnehmer, wie viele Ergebnisse relevant sind erheblich. Das Suchszenario wurde gewählt, da für Wetterberichte eine große Zahl von Webservices zur Verfügung stehen und somit eine ausreichend große Datenmenge zur Verfügung stand. Der Nachteil dieses Szenarios ist jedoch, dass alle Dienste sehr ähnliche Funktionen anbieten und entsprechend auch ähnlich beschrieben werden. Für die Suche bedeutete dies, dass es sehr viele Suchergebnisse gab, deren Relevanz auch der Suchalgorithmus kaum unterscheiden konnte.

Die letzte wesentliche Beschränkung des Tests lag im Hintergrund der Teilnehmer, die alle Wirtschaftsinformatik-Studenten mit geringen Kenntnissen in der Verwendung und der Orchestrierung von Webservices hatten. Dies könnte dazu geführt haben, dass erweiterte Funktionen gefordert wurden, die ein Endbenutzer ohne Informatikhintergrund nicht benötigen würde.

Um alle Aspekte des Systems testen zu können, wäre ein langfristiger Benutzungstest, mit einer größeren Benutzerzahl notwendig, der durchgeführte Test hat jedoch ausgereicht um einige interessante Verbesserungsvorschläge und Ideen zu liefern

6 Zusammenfassung und Ausblick

In dieser Arbeit wurde ein Webservice Suchsystem entwickelt, dass Endbenutzer bei der Suche nach für sie interessanten Diensten zur Orchestrierung unterstützt. Heutige Verfahren leisten dies aus folgenden Gründen nur bedingt:

- Das Standard-Webserviceverzeichnis „UDDI“ bietet keine ausreichenden Suchfunktionen.
- Es liegen häufig keine vollständigen Dokumentationen der Dienste durch die Dienstanbieter vor.
- Der Sprachgebrauch von Dienstanbietern und Benutzern kann sich unterscheiden.

Webbasierte Suchmaschinen für Webservices liefern bessere Ergebnisse als UDDI, allerdings lassen sie sich nur schwer in eine Orchestrierungsumgebung integrieren, und auch vorhandene Ansätze zur Verbesserung des Service Discovery sind nicht ohne weiteres auf den betrachteten Anwendungsfall übertragbar. Insbesondere Semantic Web Services, die auf den ersten Blick genau die beschriebenen Schwierigkeiten der UDDI-basierten Suche beheben, erwiesen sich als zu kompliziert in der Bedienung und Wartung.

Aus dieser Situation entstand die Idee, die in „Web 2.0“-Anwendungen häufig verwendeten Community-generierten Inhalte, sowie die vor allem durch internetbasierte Empfehlungssysteme bekannte Personalisierung als Basis für die Entwicklung eines eigenen Webservice Suchsystems zu verwenden.

Dazu wurde zuerst eine ausgiebige Literaturrecherche durchgeführt, in der die Schwierigkeiten des heutigen Service Discovery herausgearbeitet, und vorhandene Ansätze zur Verbesserung betrachtet wurden. Auf Basis dieser Betrachtungen wurde dann ein Webservice-Suchsystem entworfen, dass durch die Benutzergemeinschaft erstellte Beschreibungstexte und Tags als Datengrundlage für die Suche verwendet und dann Benutzerprofile und von den Benutzern abgegebene Relevanzbewertungen verwendet um eine personalisierte Sortierung der Suchergebnisse zu erreichen. Ein wichtiger Aspekt des Suchsystems, der es von existierenden Systemen abgrenzt, ist dass Operationen, also die Elemente der Webservices die letztendlich vom Benutzer verwendet werden, direkt als Suchergebnisse geliefert werden können. Um den Benutzern eine möglichst einfache Bedienung des Suchsystems zu ermöglichen, dienten Web-Suchmaschinen, mit deren Verwendung ein Großteil der Endbenutzer vertraut ist, als Vorlage bei der Entwicklung der

Benutzeroberfläche. Das Suchsystem wurde als Teil der SiSeOr-Orchestrierungsumgebung implementiert und in einem Benutzungstest, in dem es von acht Wirtschaftsinformatik-Studenten mit einer webbasierten Webservice-Suchmaschine verglichen wurde, evaluiert. Der Test zeigte, dass es bei verschiedenen Aspekten des Suchsystems noch Verbesserungspotential besteht, insbesondere was die Präzision der verwendeten Suchalgorithmen angeht.

Bei der Entwicklung des Suchsystems zeigte sich, dass sich community-generierte Inhalte durchaus als Basis eines solchen Suchsystems eignen, da sie eher der Perspektive anderer Benutzer entsprechen und langfristig vollständiger sind, als Beschreibungen durch Dienst-anbieter. Allerdings zeigte sich auch, dass es aufgrund des möglicherweise geringen Umfangs der einzelnen Beschreibungen schwierig ist, einen Suchalgorithmus zu verwenden, der auf gewichteten Schlagworten basiert. Diese Feststellung führte dazu, dass ein einfacher Suchalgorithmus gewählt wurde, der aus einer geringen Datenmenge eine große Zahl an Ergebnissen generiert, was zu einer Verringerung der Präzision des Suchsystems führt.

Die Zielgruppe für das entwickelte Suchsystem sind Endbenutzer ohne weitgehende Informatikkenntnisse, was ein Grund dafür war, dass einfache, natürlichsprachliche Suchanfragen verwendet werden. Bei der Evaluation wünschten sich jedoch einige Nutzer eine erweiterte Suchfunktion, in der nach speziellen Kriterien gesucht und die Suchergebnisse gefiltert werden können. Ob eine derartige Suchfunktion gewünscht wird, hängt möglicherweise von den Computerkenntnissen der Benutzer ab. Da jedoch anzunehmen ist, dass wenigstens ein Teil der späteren Benutzer über gewisse Grundkenntnisse im Informatikbereich verfügt, könnten zusätzliche Funktionen bei der Suche eine sinnvolle Ergänzung zum aktuellen Suchsystem darstellen, die optional verwendet werden kann.

Des Weiteren verfügt Seekda.com[53], das Vergleichssystem des Benutzungstests, über die Möglichkeit, Tags direkt anzuklicken, um so verwandte Dienste zu erhalten. Diese Möglichkeit lieferte aus der Sicht einzelner Benutzer sogar bessere Ergebnisse, als die Suchfunktion selbst (vgl. Kapitel 5.3). Tags sind im Internet weit verbreitet und werden als eine Möglichkeit zur einfachen semantischen Beschreibung von Inhalten betrachtet (vgl. Kapitel 2.3). In SiSeOr spielen sie jedoch nur eine kleine Rolle, da sie nur für Dienste vergeben werden können und ausschließlich zusätzlich zu den Beschreibungstexten, als Datenbasis für die vorhandene Suchfunktion verwendet werden. Die Erweiterung von SiSeOr um weitere, auf die Tags zugreifende Suchfunktionen wie z.B. anklickbare Tags in der Detail-Ansicht oder auch Tag Clouds (vgl. Kapitel 2.3), könnten die Benutzer weiter bei der Suche unterstützen.

Die Verwendung von Relevanz-Feedback und personalisierten Suchergebnissen kann genutzt werden, um die Präzision der Suchergebnisse zu erhöhen und den Benutzern an ihre Bedürfnisse angepasste Ergebnisse zu liefern. Dieser Effekt lies sich jedoch durch die Evaluation aufgrund der Beschränkungen, was zur Verfügung stehende Zeit und Teilnehmer angeht nicht überprüfen. Ein Vorgehen, wie bei im Internet verbreiteten Empfehlungssysteme, also die Empfehlung von Diensten alleine aufgrund der Informationen im Benutzerprofil sind nicht sinnvoll, da der Benutzer nicht jedes mal, wenn er eine Orchestrierungsumgebung verwendet dasselbe Ziel erreichen will und entsprechend alleine sein vergangenes Verhalten keine Aussage darüber zulässt, für welche Dienste er sich bei seiner nächsten Verwendung interessieren könnte. In Verbindung mit einem Suchalgorithmus, zur Eingrenzung und Sortierung der Suchergebnisse sind personalisierte Webservice-Empfehlungen aber durchaus sinnvoll.

Aus den Schlussfolgerungen der Arbeit ergeben sich jedoch einige Ideen, die zu einer zukünftigen Verbesserung des Suchsystems beitragen könnten. Eine offensichtliche Schwäche des Systems ist die niedrige Präzision der verwendeten Suchalgorithmen, es wäre also sinnvoll nach einer anderen Möglichkeit der Verwendung und Verarbeitung der zur Verfügung stehenden Daten zu suchen, um einen Suchalgorithmus verwenden zu können, der sowohl relevante Ergebnisse findet, als auch nicht relevante Ergebnisse besser herausfiltert als dies vom aktuellen Verfahren geleistet werden kann. Dazu könnte beispielsweise überprüft werden, ob sich Verfahren wie TF-IDF und LSI nicht doch auf die zur Verfügung stehenden Informationen übertragen lassen.

Auch im Bereich der Personalisierung gäbe es Möglichkeiten zur Verbesserung. Das Erstellen des Benutzerprofils und auch die Bewertung der Relevanz muss zur Zeit manuell von den Benutzern ausgeführt werden, während Empfehlungssysteme im World Wide Web häufig mit automatisch erfassten Informationen arbeiten. Als Ersatz für von Hand ausgefüllte Benutzerprofile könnte das „Collaborative Filtering“ (vgl. Kapitel 2.6.1) dienen, bei dem Benutzer anhand von ähnlichem Verhalten gruppiert werden. Die Schwierigkeit bei diesem Vorgehen ist, dass dazu die Verwendung der Dienste erfasst werden müsste. Alleine die Tatsache, dass ein Dienst im Orchestrierungseditor verwendet wird, lässt noch keine eindeutige Aussage über seine tatsächliche Verwendung zu, und eine Beobachtung der erzeugten Komposition zu dessen Laufzeit durch die Orchestrierungsumgebung ist schwierig. Bevor das System auf eine solche Weise verändert werden könnte muss, demnach zuerst geklärt werden, welche Art von automatischer Datenerfassung im vorliegenden Fall sinnvoll und umsetzbar ist.

Eine Einschränkung des aktuellen Systems, die bereits im Zusammenhang mit der Hierarchie der Daten beschrieben wurde, stellt die fehlende Unterstützung von

Kompositionen als Teile neuer Kompositionen dar (vgl. Kapitel 3.2.1). Eine Unterstützung von Kompositionen in SiSeOr würde nicht nur eine Verbesserung der aktuell verwendeten Ähnlichkeitssuche ermöglichen, sondern auch die Suche nach bereits erstellten Kompositionen über das hier beschriebene Suchsystem ermöglichen.

Diese Arbeit sollte klären, wie sich community-generierte Inhalte und Benutzerprofile in einen Suchalgorithmus für Webservices integrieren lassen um dadurch die Qualität der Suchergebnisse zu steigern. Das entwickelte Suchsystem stellt einen möglichen Weg für die Integration dieser Daten in einem Webservice Suchsystem dar, der trotz vorhandener Schwächen nach den Theorien, die den verwendeten Verfahren zugrunde liegen zu einer Verbesserung der Ergebnisqualität im Vergleich zu vorhandenen Systemen führen sollte. Eine endgültige Beurteilung der Qualität der Suchergebnisse würde jedoch einen langfristigen Benutzungstest mit einer echten Benutzergemeinschaft erfordern, der im Rahmen dieser Diplomarbeit nicht durchgeführt werden konnte.

Literaturverzeichnis

1: Abramowicz, W., Kaczmarek, M., Kowalkiewicz, M., Zyskowski, D., Architecture for Service Profiling*. In Proceedings of the IEEE Services Computing Workshops (September 18-22, 2006)*. IEEE Computer Society, Washington, DC, USA, 2006, S. 121-130.

2: ActiveVOS - Active Endpoints Visual Orchestration System. *http://www.activevos.com*

3: Aguilera, U. Abaitua, J., Diaz, J. et al., A Semantic Matching Algorithm for Discovery in UDDI*. In Proceedings of the International Conference on Semantic Computing(September 17-19, 2007),*. IEEE Computer Society, Washington, DC, USA, 2007, S. 751-758.

4: Aiken, D., B2C Web Service Discovery. *http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-134/aiken-positionp-wiw05.pdf*, abgerufen: 16.10.2008.

5: Amazon.de. *http://www.amazon.de*

6: Andreasen, T., Bulskov, H., Knappe, R., From Ontology over Similarity to Query Evaluation*. In R. Bernardi, M. Moortgat (Hrsg.): 2nd CoLogNET-ElsNET Symposium - Questions and Answers: Theoretical and Applied Perspectives*. , Amsterdam, Niederlande, 2003, S. 39-50.

7: Andreasen, T., Bulskov, H., Knappe, R., On Ontology-based querying*. In Stuckenschmidt, H.(Hrsg.): 18th International Joint Conference on Artificial Intelligence, Ontologies and Distributed Systems (August 9-15, 2003)*. , Acapulco, Mexico, 2003, S. 53-59.

8: Bachlechner, D., Siorpaes, K., Lausen, H., Fensel, D., Web Service Discovery - A Reality Check*. In Proceedings of the European Semantic Web Conference (ESWC)*. Budva, Montenegro, 2006.

9: Berners-Lee, T., Hendler, J., Lassila, O., The Semantic Web*. In Scientific American Magazine 284(5)*. Mai 2001, S. 34-43.

10: Bernstein, A., Kaufmann, E., Bürki, C., Klein, M., How Similar Is It? Towards Personalized Similarity Measures in Ontologies*. In Ferstl, O. K., Sinz, E. J., Eckert, S., Isselhorst, T. (Hrsg.): Wirtschaftsinformatik 2005: eEconomy, eGovernment, eSociety, 7. Internationale Tagung Wirtschaftsinformatik (22.-25 Februar 2005)*. Physica-Verlag, Bamberg, 2005, S. 1347-1366.

11: Bianchini, D., De Antonellis, V., Pernici, B., Plebani, P., Ontology-based methodology for e-service discovery*. In Information Systems 31 Nr. 4-5*. 2006, S. 361-380.

12: Brahe, S., Schmidt, K., The Story of a working workflow management system*. In Proceedings of the 2007 international ACM Conference on Supporting Group Work (Sanibel Island, Florida, USA, November 04-07, 2007)*. ACM, New York, USA, 2007, S. 249-258.

13: Burstein, M., Bussler, C., Zaremba, M.,, A Semantic Web Services Architecture*. In IEEE Internet Computing Volume 9, Issue 5*. IEEE Educational Activities Department, 2005, S. 72-81.

14: Caumanns, J., A Fast and Simple Stemming Algorithm for German Words. Technical Report, Center für Digitale Systeme, Freie Universität Berlin, 1999.

15: Deerwester, S., Dumais, S., Furnas, G., Landauer, T., Harshman, R., Indexing by Latent Semantic Analysis*. In Journal of the American Society of Information Science Vol. 41, No. 6*. 1990, S. 391-407.

16: Eclipse BPEL-Project. *http://www.eclipse.org/bpel/*

17: EuroWordNet: Building a multilingual database with wordnets for several European languages. *http://www.illc.uva.nl/EuroWordNet/*

18: Extensible Markup Language (XML). *http://www.w3.org/XML/*

19: Gamma, E., Helm, R., Johnson, R. E., *Design Patterns. Elements of Reusable Object-Oriented Software*. Addison-Wesley Longman, Amsterdam, NL, 1995.

20: Glue - Semantic Web Service Discovery. *http://swa.cefriel.it/Glue*

21: Golder, S., Huberman, B., Usage Patterns of Collaborative Tagging Systems*. In Journal of Information Science Volume 32, No. 2 (April 2006)*. 2006, S. 198-208.

22: Google.de. *http://www.google.de*

23: The GraphML File Format. *http://graphml.graphdrawing.org/*

24: Gruber, T., Ontology*. In Liu, L. , Özsu, M. T. (Hrsg.):Encyclopedia of Database Systems*. Springer-Verlag, 2008.

25: Heutschi, R., *Serviceorientierte Architektur: Architekturprinzipien und Umsetzung in die Praxis*. Springer, Berlin Heidelberg, 2007.

26: Hofmann, M., Ley, B., Dörner, C., Endbenutzergerechte Anpassung von serviceorientierten Architekturen*. In Bichler, M., Hess, T., Krcmar, H., et al. (Hrsg.): Multikonferenz Wirtschaftsinformatik, MKWI 2008 München, 26.2.2008-28.2.2008, Proceedings*. GITO-Verlag, Berlin, 2008.

27: Java Graph Visualization and Layout. *http://www.jgraph.com/*

28: JGraphT - a free Java Graph Library. *http://jgrapht.sourceforge.net/*

29: Kawamura, T., De Blasio, A., Hasegawa, M., Paolucci, M., Sycara, K., Preliminary Report of Public Experiment of Semantic Service Matchmaker with UDDI Business Registry*. In 1st International Conference on Service Oriented Computing (ICSOC 2003)*. Trento, Italy, December 2003.

30: Knappe, R., Bulskov, H., Andreasen, T., Similarity Graphs*. In Zhong, N., Ras, Z. W., Tsumoto, S., Suzuki, E. (Hrsg.): Foundations of Intelligent Systems, 14th International Symposium, ISMIS 2003, Maebashi City, Japan, October 28-31, 2003, Proceedings*. Springer, 2003, S. 668-672.

31: Kolbitsch, J., Marer, H., The Transformation of the Web: How Emerging Communities Shape the Information we Consume*. In Journal of Universal Computer Science* . 2005, S. 187-213.

32: Kuck, J., Gnasa, M., Context-sensitive service discovery meets information retrieval*. In Proceedings of the fifth IEEE International Conference on Pervasive Computing and Communications Workshops (March 19-23, 2007)*. IEEE Computer Society, Washington, DC, USA, 2007, S. 601-605.

33: Kuck, J., Reichartz, F., A collaborative and featurebased approach to Context-Sensitive Service Discovery*. In 16th International World Wide Web Conference, 5th WWW Workshop on Emerging Applications for Wireless and Mobile Access*. Banff, Alberta, Canada, 2007.

34: Lieberman, H., Paternó, F., Klann, M., Wulf, V., End-User Development: An Emerging Paradigm. *In H. Lieberman; F. Paternó; V. Wulf (Hrsg.): End User Development*. Springer, Dordrecht, 2006, S. 9-16.

35: Liu, X., Huang, G., Mei, H., Towards End User Service Composition. *In Proceedings of the 31st Annual International Computer Software and Applications Conference, Volume 01 (July 24-27, 2007)*. IEEE Computer Society, Washington, DC, USA, 2007, S. 676-678.

36: McAffer, J., Lemieux, J.-M., *Eclipse Rich Client Platform: Designing, Coding and Packaging Java Applications*. Addison Wesley Professional, Amsterdam, Niederlande, 2005.

37: Microsoft, UBR Shutdown FAQ. *http://uddi.microsoft.com/about/FAQshutdown.htm*, abgerufen 19.09.2008.

38: Middleton, S., Shadbolt, N., De Roure, D., Ontological User Profiling in Recommender Systems. *In ACM Transactions on Information Systems 22, 1 (Jan. 2004)*. ACM, New York, USA, 2004, S. 54-88.

39: Mobasher, B., Recommender Systems. *In Künstliche Intelligenz, Special Issue on Web Mining No. 3*. BöttcherIT Verlag, Bremen, 2007, S. 41-43.

40: Movielens Movie Recommendations. *http://www.movielens.org*

41: (N)Onliner-Atlas - Deutschlands größte Studie Nutzung und Nicht-Nutzung des Internets. *http://www.initiatived21.de/fileadmin/files/08_NOA/NONLINER2008.pdf*

42: O'Reilly, T., What Is Web 2.0. *http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html* , 2005, abgerufen: 12.10.2008.

43: OWL Web Ontology Language. *http://www.w3.org/TR/owl-features/*

44: OWL-S: Semantic Markup for Web Services. *http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/*

45: Paczynski, F. Diplomarbeit: Vereinfachte Modifikation serviceorientierter Software durch Abstraktion und Transformation am Beispiel von Web Services. Universität Siegen, 2008

46: Paolucci, M., Kawamua, T., Payne, T., Sycara, K., Importing the Semantic Web in UDDI. *In Bussler, C., Hull, R., McIlraith, A., et al. (Hrsg.): Revised Papers From the international Workshop on Web Services, E-Business and the Semantic Web (May 27-28, 2002)*. Springer-Verlag, London, May 2002, S. 225-236.

47: Paolucci, M., Kawamura, T., Payne, T., Sycara, K., Semantic Matching of Web Service Capabilities. *In Horrocks, I., Hendler, J.A. (Hrsg.): Proceedings of the First International Semantic Web Conference (June 09-12, 2002)*. London, 2002.

48: Pipek, V., Kahler, H., Supporting Collaborative Tailoring. *In H. Lieberman; F. Paternó; V. Wulf (Hrsg.): End User Development*. Springer, Dordrecht, 2006, S. 315-345.

49: ProgrammableWeb.com. *http://www.programmableweb.com*

50: Reichling, T., Veith, M., Wulf, V., Expert Recommender: Designing for a Network Organization. *In Computer Supported Cooperative Work: The Journal of Collaborative Computing*. 2007, S. 431-465.

51: Resource Description Framework (RDF). *http://www.w3.org/RDF/*

52: Salton, G., Buckley C., Term Weighting Approaches in Automatic Text Retrieval. *In Technical Report TR-87-881* . Cornell University, Ithaca, NY, USA, 1987.

53: seekda! - Web Service Search Engine. *http://seekda.com/*

54: Shanfeng, Z., Xiaotie, D., Kang, C., Weimin, Z., Using Online Relevance Feedback to Build Effective Personalized Metasearch Engine. *In Proceedings of the Second international Conference on Web information Systems Engineering (Wise'01), Volume 1 (December 03-06, 2001)*. IEEE Computer Society, Washington, DC, USA, 2001, S. 262-268.

55: Shi, X., The Challenge of Semantic Web Services. *In IEEE Intelligent Systems vol 23, no. 2*. IEEE Computer Society, Washington, DC, USA, 2008, S. 5.

56: SOAP Specifications. *http://www.w3.org/TR/soap/*

57: Su L. T., A comprehensive and systematic model of user evaluation of web search engines: I. theory and background. *In Journal Of The American Society Of Information Science And Technology 54, 13 (Nov. 2003)*. John Wiley & Sons, Inc, New York, NY, USA, 2003, S. 1175-1192.

58: Su L. T., A comprehensive and systematic model of user evaluation of web search engines: II. an evaluation by undergraduates. *In Journal Of The American Society Of Information Science And Technology 54, 13 (Nov. 2003)*. John Wiley & Sons, Inc, New York, NY, USA, 2003, S. 1193-1223.

59: Su, L. T., A comprehensive and systematic model of user evaluation of web search engines: I. theory and background. *In Journal Of The American Society Of Information Science And Technology 54, 13 (Nov. 2003)*. John Wiley & Sons, Inc, New York, NY, USA, 2003, S. 1175-1192.

60: OASIS UDDI Specifications TC. *http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm*

61: van Rijsbergen, C.J., Robertson, S.E., Porter, M.F., New models in probabilistic information retrieval. *In British Library Research and Development Report, no. 5587, cap. 6* . London, 1980.

62: Web Services Business Execution Language Version 2.0. *http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html*

63: Wen, Q., He, J., Personalized Recommendation Services Based on Service-Oriented Architecture. *In Proceedings of the 2006 IEEE Asia-Pacific Conference on Service Computing (December 12-15, 2006)*. IEEE Computer Society, Washington, DC, USA, 2006, S. 356-361.

64: Wikipedia - Die freie Enzyklopädie. *http://de.wikipedia.org*

65: WordNet - a lexical database for the English language. *http://wordnet.princeton.edu/*

66: WS-BPEL Extension for people. *http://www.ibm.com/developerworks/webservices/library/specification/ws-bpel4people/*

67: Web Service Description Language (WSDL). *http://www.w3.org/TR/wsdl*

68: WVTool - The Word & Web Vector Tool. *http://www.wvtool.nemoz.org*

69: XSL Transformations (XSLT) Version 1.0. *http://www.w3.org/TR/xslt*

70: yEd Graph Editor. *http://www.yworks.com/en/products_yed_about.html*

Abbildungsverzeichnis

Abbildung 2.1: SOA-Interaktionsdreieck ([25]) 5
Abbildung 2.2: Suchmaske Glue-Demo „Purchase Goal“ [20].....13
Abbildung 2.3: Tag Cloud auf Seekda.com[53].....16
Abbildung 2.4: Beispiel für einen Ähnlichkeitsgraphen [6].....22
Abbildung 2.5:Einfacher Ähnlichkeitsgraph [6] 23
Abbildung 2.6: Spezifitätskosten in Ähnlichkeitsgraphen.....24
Abbildung 2.7: Personalisierte Empfehlungen bei Amazon.de[5].....27
Abbildung 3.1: Beispiel einer Web-Suchmaschine (Google.de[22]).....31
Abbildung 3.2: Relevanz-Feedback bei Amazon.de[5].....34
Abbildung 3.3: Aufbau des Suchsystems.....35
Abbildung 3.4: Beispiele für unterschiedlichen Umfang von Dienstbeschreibungen.....38
Abbildung 3.5: Hierarchie der verwendeten Daten.....40
Abbildung 3.6: Beispiel Dienstgraph.....42
Abbildung 3.7: Beispiel: Dienstgraph.....47
Abbildung 3.8: Beispiel: Ähnlichkeitsbeziehungen zwischen Knoten.....49
Abbildung 4.1: WSDL-Verarbeitung in EUSOP.....62
Abbildung 4.2: EUSOP – End User Service Orchestration Platform [26].....62
Abbildung 4.3: Die SiSeOr-Benutzeroberfläche.....63
Abbildung 4.4: Anbindung Suchsystem an EUSOP/SiSeOr.....65
Abbildung 4.5: Beispiel: Dienstbeschreibungen auf Seekda.com[53].....67
Abbildung 4.6: Beispiel: Dienstgraph mit zwei Diensten.....69
Abbildung 4.7: Beispiel: Ähnlichkeitsgraph.....70
Abbildung 4.8: Beispiel: Auszug GraphML des Servicegraphen.....71
Abbildung 4.9: Beispiel: Auszug GraphML des Ähnlichkeitsgraphen.....73
Abbildung 4.10: Beispiel: XML-Datei mit einzelnem Benutzerprofil.....73
Abbildung 4.11: Beispiel: XML-Datei mit Benutzerfeedback.....74
Abbildung 4.12: Benutzerverwaltung.....77
Abbildung 4.13: Suchergebnisse.....78
Abbildung 4.14: Fortsetzung Suchergebnisse.....78
Abbildung 5.1: Dienstbeschreibung auf Seekda.com[53].....82

Tabellenverzeichnis

Tabelle 3.1: Anforderungen an das Suchsystem.....34
Tabelle 3.2: Beispiel: Ausgangssituation.....46
Tabelle 3.3: Beispiel: Ähnlichkeitsmatrix.....48
Tabelle 3.4: Beispiel: Berechnete Suchergebnisse.....50
Tabelle 3.5: Beispiel: Sortierte Ergebnisliste.....51
Tabelle 3.6: Beispiel: Einfache Benutzerprofile.....53
Tabelle 3.7: Beispiel: Relevanzbewertungen.....54
Tabelle 3.8: Beispiel: Erfassung der Relevanzbewertungen.....54
Tabelle 3.9: Beispiel: Suchergebnisse für Benutzer1/Benutzer2.....57
Tabelle 3.10: Beispiel: Suchergebnisse für Benutzer3/Benutzer4.....58

Anhang: Fragebögen Benutzungstest

Auf den folgenden Seiten befinden sich die ausgefüllten Fragebögen aus dem Benutzungstest. Die Fragebögen sind nach Teilnehmern sortiert, und pro Teilnehmer in der Reihenfolge, in der sie ausgefüllt wurden. Während des Tests hat jeder Teilnehmer nacheinander die beiden Suchsysteme benutzt, und dabei jeweils einen Fragebogen ausgefüllt („Search Session Questionaire“), der sich mit der Funktionalität und Benutzerfreundlichkeit des jeweiligen Systems befasst. Daraufhin wurde ein weiterer Fragebogen ausgefüllt („Post-Session Questionaire“), der die beiden Systeme miteinander vergleicht und die allgemeine Zufriedenheit mit den gelieferten Ergebnissen bewertet. Die Fragebögen basieren auf den von L.T. Su [59] genutzten und liegen in englischer Sprache vor.

Search Session Questionnaire

1. Participant number: 1
2. Name of search engine: SiSeOr
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 2:00 p.m.
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - ☒ Search strategy # 1 weather forecast
   - ☒ Search strategy # 2 weather forecast international towns
   - ☒ Search strategy # 3 world weather cities
   - ☒ Search strategy # 4 world weather week international
   - ☒ Search strategy # 5 international weather forecast city
5. How many of the first 20 results of the best strategy are...
   relevant 1
   partially relevant 10
   not relevant 9
6. Finishing time (When above tasks are completed): 2:27 p.m.
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ◉ 7 Extremely satisfactory
      Reasons: very quick, less than 1 second

   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine?Extremely unsatisfactory ○ 1 ◉ 2 ○ 3 ○ 4 ○ 5 ◉ 6 ○ 7 Extremely satisfactory
      Reasons: there should be a "Go" button to start a search also by a mouse click (perform), combo boxes should contain the first item

   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ◉ 7 Extremely satisfactory
      Reasons: useful details of service information, list of operations etc.

   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ◉ 6 ○ 7 Extremely satisfactory
      Reasons: result list box should be bigger, placement of search window is not well chosen / ~~bad~~ window too small (can it be resized?

   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ◉ 6 ○ 7 Extremely satisfactory
      Reasons: there were some results (about 10% of total results) that did not fit at all to my search

Search Session Questionnaire

1. Participant number: 1
2. Name of search engine: Seeda.com
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 2:45 p.m.
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - ☐ Search strategy # 1 international weather forecast city
   - ☐ Search strategy # 2 World weather
   - ☐ Search strategy # 3 world weather city
   - ☐ Search strategy # 4 world weather week
   - ☒ Search strategy # 5 weather forecast
5. How many of the first 20 results of the best strategy are...
   relevant 4
   partially relevant 13
   not relevant 3
6. Finishing time (When above tasks are completed): 3:02 p.m.
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ◉ 7 Extremely satisfactory
      Reasons: very quick system

   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine?Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
      Reasons: the search text box + button should always on top into the centre frame, whatever which site is opened

   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ◉ 7 Extremely satisfactory
      Reasons: details of service by one-click (very useful), country is mentioned → helps to distuingish web services with same title & different country, Google Ads etc

   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ◉ 7 Extremely satisfactory
      Reasons: good placement of search results

   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
      Reasons: some results that did not fit at all (about 5%) to my search

1

Post-Session Questionnaire

1. Searcher's judgment concerning precision ratio. Please indicate your degree of satisfaction with the proportion of relevant services using a 7-point scale, 1 representing extremely unsatisfactory, and 7, extremely satisfactory.

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
   Reasons: ~~good way to~~ nice tool to look for web services

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
   Reasons: good way to perform a better search by adding web service results to a simple google search

2. Time saving. Does using the engine to find services save you time?

   SiSeOr-Searchsystem:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ⊗ 7 Saving me a lot of time

   seekda.com:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ⊗ 7 Saving me a lot of time

3. Searcher's judgment concerning value of search results as a whole. Taking the set of search results with summaries and other information, when available, as a whole, its value or usefulness in meeting your need or resolving your problem is:

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
   Reasons: some results did not fit at all

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
   Reasons: see above

4. Overall performance (SUCCESS). Considering your experience in using the engine to retrieve information for your information problem, how would you rate the overall success of of the engine in providing help for your information need or problem?

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
   Reasons: some small things to improve, but already a nice plugin to search Le5 services!

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ⊗ 7 Extremely satisfactory
   Reasons: there is practically no improvement necessary

Thank you for your participation in this research project.

Search Session Questionnaire

1. Participant number: 2
2. Name of search engine: Eclipse Plugin
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): [illegible]
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - ☒ Search strategy # 1 Suchwörter: weather forecast city
   - ☒ Search strategy # 2 Suchwörter: weather forecast
   - ☐ Search strategy # 3 Suchwörter: weather
   - ☐ Search strategy # 4 _____
   - ☐ Search strategy # 5 _____
5. How many of the first 20 results of the best strategy are...
   relevant 17
   partially relevant 3
   not relevant ____
6. Finishing time (When above tasks are completed): _____
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:
   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: _____
   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine? Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: _____
   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: Nicht vorhanden
   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: ~~Keinte etwas selbst~~ Wenn eine Hilfe vorhanden wäre wäre es ok, da nicht immer alles selbsterklärend ist
   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: _____

Search Session Questionnaire

1. Participant number: 2
2. Name of search engine: seekda.com
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 15:25
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - ☐ Search strategy # 1 Stichwort: ~~wetter~~ weather
   - ☐ Search strategy # 2 Stichwort: ~~wetter~~ weather forecast
   - ☐ Search strategy # 3 Erweiterte Suche: wetter, Germany
   - ☐ Search strategy # 4 _____
   - ☐ Search strategy # 5 _____
5. How many of the first 20 results of the best strategy are...
   relevant 13 ~~IIII IIII IIII II~~
   partially relevant 2 ~~II~~
   not relevant ~~3~~
   keine Angabe 5 ~~IIII~~
6. Finishing time (When above tasks are completed): 15:35
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ⊗ 7 Extremely satisfactory
      Reasons: _____

   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine? Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ⊗ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: _____

   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ⊗ 7 Extremely satisfactory
      Reasons: _____

   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ⊗ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: Hotels Monitor wird keine Beschreibung geliefert etc. diese Dienste werden angezeigt

   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
      Reasons: _____

2

Post-Session Questionnaire

1. Searcher's judgment concerning precision ratio. Please indicate your degree of satisfaction with the proportion of relevant services using a 7-point scale, 1 representing extremely unsatisfactory, and 7, extremely satisfactory.

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ⊗ 7 Extremely satisfactory
   Reasons: Bei allen Diensten wird auch eine Beschreibung vorhanden

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ⊗ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

2. Time saving. Does using the engine to find services save you time?

   SiSeOr-Searchsystem:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ⊗ 5 ○ 6 ○ 7 Saving me a lot of time

   seekda.com:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Saving me a lot of time

3. Searcher's judgment concerning value of search results as a whole. Taking the set of search results with summaries and other information, when available, as a whole, its value or usefulness in meeting your need or resolving your problem is:

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ⊗ 7 Extremely satisfactory
   Reasons:_____

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
   Reasons:_____

4. Overall performance (SUCCESS). Considering your experience in using the engine to retrieve information for your information problem, how would you rate the overall success of of the engine in providing help for your information need or problem?

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
   Reasons:_____

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ⊗ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

Thank you for your participation in this research project.

Search Session Questionnaire

1. Participant number: 3
2. Name of search engine: seekda.com
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 14 13
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - ☐ Search strategy # 1 wheather, outlook eingegeben
   - ☐ Search strategy # 2 wheather eingegeben
   - ☐ Search strategy # 3 wheather information eingegen
   - ☐ Search strategy # 4 Advanced search benutzt: wheather tag: global
   - ☒ Search strategy # 5 wheather + Advanced search tag: forecast

   #6 wheather, forecast eingegeben
5. How many of the first 20 results of the best strategy are...
   relevant 4?
   partially relevant 3
   not relevant 13
6. Finishing time (When above tasks are completed): 14 34
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ☒ 6 ○ 7 Extremely satisfactory
      Reasons: —

   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine?Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ☒ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: überfüllt, schnell unübersichtlich

   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ☒ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: Wo finde ich FAQ z.B. ?? – Help nicht im Blickfeld! man muss erst nach "suchen"

   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ☒ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: Bei den Abstracts schlecht unterscheidbar zwischen wichtig und nicht

   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ☒ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: Man muss schon ganz genau wissen, wonach man suchen will; Annäherung an in meinen Augen nicht möglich

Search Session Questionnaire

1. Participant number: 3
2. Name of search engine: SiSeOr
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 15^00
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - ☐ Search strategy # 1 weather eingegeben
   - ☐ Search strategy # 2 weather, forecast eingegeben
   - ☒ Search strategy # 3 weather, forecast, global eingegeben
   - ☐ Search strategy # 4 _____
   - ☐ Search strategy # 5 _____
5. How many of the first 20 results of the best strategy are...
   relevant 7?
   partially relevant /
   not relevant 13
6. Finishing time (When above tasks are completed): 15^10
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ☒ 6 ○ 7 Extremely satisfactory
      Reasons: —

   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine? Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ☒ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: übersichtlich, obwohl Suchergebnisfenster zu klein ist
      wo wird angezeigt, wieviele Suchergebnisse? Gibt es sowas?

   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ☒ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: wo ist so etwas? FAQ? Hilfe, etc?

   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ☒ 3 ○ 4 ☒ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: abstracts gut ~~wo ist der Link? Hilfe?~~ number of hits? wo ist sowas? Suchergebnisliste + Bewertungsleiste sind sehr gut

   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ☒ 6 ○ 7 Extremely satisfactory
      Reasons: —

3

Post-Session Questionnaire

1. Searcher's judgment concerning precision ratio. Please indicate your degree of satisfaction with the proportion of relevant services using a 7-point scale, 1 representing extremely unsatisfactory, and 7, extremely satisfactory.

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: Bewertung ist Hilfe

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: Relevanz lässt sich nicht entscheiden (für mich)

2. Time saving. Does using the engine to find services save you time?

   SiSeOr-Searchsystem:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Saving me a lot of time

   seekda.com:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Saving me a lot of time

3. Searcher's judgment concerning value of search results as a whole. Taking the set of search results with summaries and other information, when available, as a whole, its value or usefulness in meeting your need or resolving your problem is:

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: obwohl hier nur 4 in meinen Augen brauchbar sind sind diese Ergebnisse trotzdem besser als von seekda.com

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: S.O.

4. Overall performance (SUCCESS). Considering your experience in using the engine to retrieve information for your information problem, how would you rate the overall success of of the engine in providing help for your information need or problem?

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: —

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: war mir persönlich zu unübersichtlich und ich konnte nicht in der Relevanz unterscheiden!

Thank you for your participation in this research project.

Search Session Questionnaire

1. Participant number: 4
2. Name of search engine: SiSeOs
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 14:42
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - ☒ Search strategy # 1 Suchtab mit Stichwort Weather
   - ☐ Search strategy # 2 Google Web tab
   - ☐ Search strategy # 3 Suchtab mit Stichwort travel
   - ☐ Search strategy # 4 _____
   - ☐ Search strategy # 5 _____
5. How many of the first 20 results of the best strategy are...
   relevant 6
   partially relevant 2
   not relevant 12
6. Finishing time (When above tasks are completed): 14:54
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: feels like just in time

   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine?Extremely unsatisfactory ○ 1 ☒ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: 2 search tabs and web service searches are confusing

   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: not needed, but also no context help when searching for it

   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ☒ 6 ○ 7 Extremely satisfactory
      Reasons: some more informations would be nice, but also it is nice that the search output is not confusing

   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ☒ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: the search engine did not understand me

Search Session Questionnaire

1. Participant number: 4
2. Name of search engine: seekda.com
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 15:22
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - ☐ Search strategy # 1 Weather, Keyword
   - ☐ Search strategy # 2 ● Tag Cloud
   - ☐ Search strategy # 3 Travel keyword
   - ☒ Search strategy # 4 Most used → liefert 1 funktionieren Service
   - ☐ Search strategy # 5 ↳ alle Services waren verfügbar
5. How many of the first 20 results of the best strategy are...
   relevant 4
   partially relevant /
   not relevant 16
6. Finishing time (When above tasks are completed): 15:32
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ☒ 6 ○ 7 Extremely satisfactory
      Reasons: feels like just on time

   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine?Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ☒ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: some kind of information overload

   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ☒ 6 ○ 7 Extremely satisfactory
      Reasons: not needed to solve the problem, but available and well written

   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ☒ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: Information overload

   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ☒ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: I found in the end what I was searching for, but it was a bit like trail and error. Not a real search process

4

Post-Session Questionnaire

1. Searcher's judgment concerning precision ratio. Please indicate your degree of satisfaction with the proportion of relevant services using a 7-point scale, 1 representing extremely unsatisfactory, and 7, extremely satisfactory.

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

2. Time saving. Does using the engine to find services save you time?

   SiSeOr-Searchsystem:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Saving me a lot of time

   seekda.com:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Saving me a lot of time

3. Searcher's judgment concerning value of search results as a whole. Taking the set of search results with summaries and other information, when available, as a whole, its value or usefulness in meeting your need or resolving your problem is:

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

4. Overall performance (SUCCESS). Considering your experience in using the engine to retrieve information for your information problem, how would you rate the overall success of of the engine in providing help for your information need or problem?

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

Thank you for your participation in this research project.

Search Session Questionnaire

1. Participant number: 5
2. Name of search engine: Seekda
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 2:18 PM
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - ☐ Search strategy # 1 weather forecast barcelona (0)
   - ☒ Search strategy # 2 weather forecast (52)
   - ☐ Search strategy # 3 -> advanced search: weather forecast country: de
   - ☐ ~~Search strategy # 4~~ tag: weather (0)
   - ☐ Search strategy # 5 _____
5. How many of the first 20 results of the best strategy are...
   relevant 19
   partially relevant 1
   not relevant ____
6. Finishing time (When above tasks are completed): 2:27
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ⊗ 7 Extremely satisfactory
      Reasons: Query results show up in no time, but response time is not explicitly indicated (unlike google...)
   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine? Extremely unsatisfactory ○ 1 ○ 2 ⊗ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: Standart search ok - but advanced search has imo uncommon fields, such as Provider. ~~Also the advanced options would be~~
   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
      Reasons: ok - but help button links to faqs, so why don't label the button FAQs?
   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ⊗ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: nice "use now" feature for other services, but search results should get more space, compared to "Ads" and "recent news"
   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ○ 2 ⊗ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: news and ads are too much, certainly useful for finding services for re-use quickly

Search Session Questionnaire

1. Participant number: 5
2. Name of search engine: GiSeOr
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 2.09
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - [x] Search strategy # 1 weather forecast
   - [ ] Search strategy # 2 " " local
   - [ ] Search strategy # 3 wetter vorhersage (0)
   - [ ] Search strategy # 4 weather
   - [ ] Search strategy # 5 _____
5. How many of the first 20 results of the best strategy are...
   relevant 19
   partially relevant ____
   not relevant ____
6. Finishing time (When above tasks are completed): 2:14
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ⊗ 7 Extremely satisfactory
      Reasons: quick

   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine? Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
      Reasons: details are nicely structured, operations, if available can easily be extracted.

   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ⊗ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: Didn't find a help for the search feature but didn't lead to need one.

   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
      Reasons: Suchergebnisse could use a longer column. detail infos in seperate area are nice.

   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ⊗ 7 Extremely satisfactory
      Reasons: _____

5

Post-Session Questionnaire

1. Searcher's judgment concerning precision ratio. Please indicate your degree of satisfaction with the proportion of relevant services using a 7-point scale, 1 representing extremely unsatisfactory, and 7, extremely satisfactory.

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
   Reasons: well structured, all infos / meta data can be extracted easily, user can see at once if a certain operation is available.
   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ⊗ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: infos on operations are kind of hidden within the textual search results.

2. Time saving. Does using the engine to find services save you time?

   SiSeOr-Searchsystem:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ⊗ 7 Saving me a lot of time

   seekda.com:
   No time saving at all ○ 1 ⊗ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Saving me a lot of time

3. Searcher's judgment concerning value of search results as a whole. Taking the set of search results with summaries and other information, when available, as a whole, its value or usefulness in meeting your need or resolving your problem is:

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ⊗ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: 'Informationen zum Service' sehr übersichtlich!

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ⊗ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: _____

4. Overall performance (SUCCESS). Considering your experience in using the engine to retrieve information for your information problem, how would you rate the overall success of of the engine in providing help for your information need or problem?

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
   Reasons: _____

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ⊗ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: _____

Thank you for your participation in this research project.

Search Session Questionnaire

1. Participant number: 6
2. Name of search engine: seekda.com
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 14:35
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - [x] Search strategy # 1 None → "featured" box → Global Weather
   - [x] Search strategy # 2 international weather → 10 results
   - [x] Search strategy # 3 tag: weather → 13 results
   - [ ] Search strategy # 4 _____
   - [ ] Search strategy # 5 _____
5. How many of the first 20 results of the best strategy are...
   relevant 5
   partially relevant 3
   not relevant 5
6. Finishing time (When above tasks are completed): 14:45
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
      Reasons: quite fast for US-website

   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine? Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ⊗ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: small and simple, could offer more fields in advanced mode

   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ○ 2 ⊗ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: none for Consumers, BUT: none was neccessary! it just works.

   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
      Reasons: not overloaded, could provide more info though.

   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ○ 7 Extremely satisfactory
      Reasons: google-like: small, fast, simple

Search Session Questionnaire

1. Participant number: 6
2. Name of search engine: asseach / SiseOk
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 15:20
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - ☒ Search strategy # 1 Suche: weather international
   - ☐ Search strategy # 2 ~~Suche: Suche: weather~~
   - ☐ Search strategy # 3 Suche: weather city international
   - ☐ Search strategy # 4 Suche: weather
   - ☐ Search strategy # 5 _____
5. How many of the first 20 results of the best strategy are...
   relevant 3
   partially relevant 4
   not relevant 13
6. Finishing time (When above tasks are completed): 15:35
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ☒ 7 Extremely satisfactory
      Reasons: seems do cache.

   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine?Extremely unsatisfactory ○ 1 ☒ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: no advanced search, limited resultset -window-size

   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ☒ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: none?

   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ☒ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: links didn't open, badly structured

   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ☒ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: well to use, fast, simple

6

Post-Session Questionnaire

1. Searcher's judgment concerning precision ratio. Please indicate your degree of satisfaction with the proportion of relevant services using a 7-point scale, 1 representing extremely unsatisfactory, and 7, extremely satisfactory.

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

2. Time saving. Does using the engine to find services save you time?

   SiSeOr-Searchsystem:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Saving me a lot of time

   seekda.com:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Saving me a lot of time

3. Searcher's judgment concerning value of search results as a whole. Taking the set of search results with summaries and other information, when available, as a whole, its value or usefulness in meeting your need or resolving your problem is:

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

4. Overall performance (SUCCESS). Considering your experience in using the engine to retrieve information for your information problem, how would you rate the overall success of of the engine in providing help for your information need or problem?

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

Thank you for your participation in this research project.

Search Session Questionnaire

1. Participant number: 7
2. Name of search engine: Si:So
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 14:15
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - [x] Search strategy # 1 Suchwort: weather ⇒ Get Weather By City State (1)
   - [ ] Search strategy # 2 weather city ⇒ Get Five Day Forecast By ZIP (1)
   - [ ] Search strategy # 3 _____
   - [ ] Search strategy # 4 _____
   - [ ] Search strategy # 5 _____
5. How many of the first 20 results of the best strategy are...
   relevant 11
   partially relevant 𝍸 ||||
   not relevant 4
6. Finishing time (When above tasks are completed): 14:25
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ⊗ 7 Extremely satisfactory
      Reasons: instant response

   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine? Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ⊗ 6 ⊗ 7 Extremely satisfactory
      Reasons: kein Button für Dumme, sonst simpel

   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ○ 2 ⊗ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: außer Tooltips im Query Result nichts gefunden

   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ⊗ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: Operation Dienst in Klammern hinter Name am Anfang nicht wahrgenommen
      - beim Query Wide auf 2 Suchergebnis passiert nichts ⇒ mehr Informationen über gefundene Service?
      - sonst relativ übersichtlich

   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ⊗ 7 Extremely satisfactory
      Reasons: _____

Search Session Questionnaire

1. Participant number: 7
2. Name of search engine: seekda.com
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 14:35 (35)
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - ☐ Search strategy # 1 Suchanfrage: "weather" -> Diensl: Ubelbar im neue Tab -> Use Now -> Get Weather For City State
   - ☐ Search strategy # 2 _____
   - ☐ Search strategy # 3 _____
   - ☐ Search strategy # 4 _____
   - ☐ Search strategy # 5 _____
5. How many of the first 20 results of the best strategy are...
   relevant IIIII
   partially relevant 10 (doppelte in jew. Kategorie gezählt)
   not relevant 3
6. Finishing time (When above tasks are completed): 14:47
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: ziemlich schnell

   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine?Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: Buttons, Advanced Search, Help bei Adv. Search, dezent, simpel

   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: außer die Tipps bei Adv. Search nichts gefunden

   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: _____

   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: _____

(7)

Post-Session Questionnaire

1. Searcher's judgment concerning precision ratio. Please indicate your degree of satisfaction with the proportion of relevant services using a 7-point scale, 1 representing extremely unsatisfactory, and 7, extremely satisfactory.

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

2. Time saving. Does using the engine to find services save you time?

   SiSeOr-Searchsystem:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Saving me a lot of time

   seekda.com:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Saving me a lot of time

3. Searcher's judgment concerning value of search results as a whole. Taking the set of search results with summaries and other information, when available, as a whole, its value or usefulness in meeting your need or resolving your problem is:

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons:_____

4. Overall performance (SUCCESS). Considering your experience in using the engine to retrieve information for your information problem, how would you rate the overall success of of the engine in providing help for your information need or problem?

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: sofortige Anzeige von Diensten und Operationen, aber im Suchergebnis-Anzeige ist diese Differenzierung nicht sofort klar

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: Listet nur Services: erst Service auswählen und auf die New Slide um Operation zu sehen → nicht intuitiv + umständlich

Thank you for your participation in this research project.

Search Session Questionnaire

1. Participant number: 8
2. Name of search engine: seekda.com
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 14:16
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - ☐ Search strategy # 1 weather service
   - ☒ Search strategy # 2 weather service world
   - ☐ Search strategy # 3 weather service global
   - ☐ Search strategy # 4 _____
   - ☐ Search strategy # 5 _____
5. How many of the first 20 results of the best strategy are...
   relevant 7
   partially relevant 11
   not relevant 2
6. Finishing time (When above tasks are completed): 14:27
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: immediatly response

   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine?Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: easy to use; Google like including adv. search

   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: not used as the interface is self expaining but information is present

   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: overview is not good as the description is ~~not~~ to short to understand the service. Detail view is good

   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: _____

Search Session Questionnaire

1. Participant number: 8
2. Name of search engine: Fiso
3. Starting time (When you are connected to the search engine, e.g., 3:45 p.m.): 14:38
4. Search strategies:
   Please start searching on your topic. You must use at least three different strategies and record them as you go along. Choose the best strategy for your search by evaluating the relevancy of search results from each strategy. Check the box of the strategy that produces the best results.
   - ☐ Search strategy # 1 Weather service
   - ☒ Search strategy # 2 Weather service world
   - ☐ Search strategy # 3 weather service global
   - ☐ Search strategy # 4 _____
   - ☐ Search strategy # 5 _____
5. How many of the first 20 results of the best strategy are...
   relevant 1
   partially relevant 10
   not relevant 9
6. Finishing time (When above tasks are completed): 14:45
7. Evaluation of features of the search engine:
   Please give a satisfaction rating for each of the following system features and interaction and indicate your reasons for the particular rating:

   a. Response time
      How satisfied are you with the time taken by the engine from the submission of a search command till the display of search results in response to your query?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: Fast response

   b. Search interface
      How satisfied are you with the search forms and commands provided by the engine?Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: clean interface, easy to use no advanced search

   c. Online documentation
      How satisfied are you with online search help and instructions (e.g., database description, search tips, help, or FAQs)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: not available (not found)

   d. Output display: How satisfied are you with the format and components of search output (e.g., abstract, URL, number of hits, relevance score, and hyperlinks, etc.)?
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: good overview incl. description, nice rating bars, not ordered by relevance, direction can not be changed,

   e. Interaction
      How satisfied are you with the information exchange between you and the search engine in general or in particular related to some specific aspect of the search process? no detail
      Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
      Reasons: good, as the search tool is integrated in the applicat

(8)

Post-Session Questionnaire

1. Searcher's judgment concerning precision ratio. Please indicate your degree of satisfaction with the proportion of relevant services using a 7-point scale, 1 representing extremely unsatisfactory, and 7, extremely satisfactory.

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: relevant services are not on the top of the list

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: good selection of services with high relevance

2. Time saving. Does using the engine to find services save you time?

   SiSeOr-Searchsystem:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Saving me a lot of time

   seekda.com:
   No time saving at all ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Saving me a lot of time

3. Searcher's judgment concerning value of search results as a whole. Taking the set of search results with summaries and other information, when available, as a whole, its value or usefulness in meeting your need or resolving your problem is:

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: no detail information, good overview

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: overview not as good as SiSeOr, but a very good detail-view.

4. Overall performance (SUCCESS). Considering your experience in using the engine to retrieve information for your information problem, how would you rate the overall success of of the engine in providing help for your information need or problem?

   SiSeOr-Searchsystem:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: does not filter unrelevant results when adding additional keywords

   seekda.com:
   Extremely unsatisfactory ○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 Extremely satisfactory
   Reasons: good multiple based on the query

Thank you for your participation in this research project.

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, insbesondere keine anderen als die angegebenen Informationen aus dem Internet.

Diejenigen Paragraphen der für mich gültigen Prüfungsordnung, welche etwaige Betrugsversuche betreffen, habe ich zur Kenntnis genommen.

Der Speicherung meiner Diplomarbeit zum Zweck der Plagiatsprüfung stimme ich zu. Ich versichere, dass die elektronische Version mit der gedruckten Version inhaltlich übereinstimmt.

Siegen, 13.02.2009

_____

Thorsten Theelen