Karlsruhe Institute of Technology

Aufbau und Konsolidierung einer Konzepthierarchie für Anforderungsbeschreibungen aus unterschiedlichen Wissensquellen

Bachelorarbeit
von

Maximilian Wessendorf

An der Fakultät für Informatik
Institut für Programmstrukturen
und Datenorganisation (IPD)

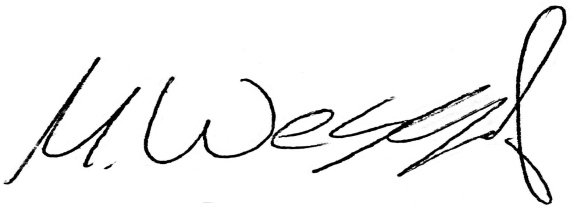
Erstgutachter:	Prof. Dr. Walter F. Tichy
Zweitgutachter:	Prof. Dr. Ralf H. Reussner
Betreuender Mitarbeiter:	Tobias Hey, M.Sc.

Bearbeitungszeit: 12.06.2019 – 11.12.2019

Ich versichere wahrheitsgemäß, die Arbeit selbstständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde.

Die Regeln zur Sicherung guter wissenschaftlicher Praxis im Karlsruher Institut für Technologie (KIT) habe ich befolgt.

Karlsruhe, 11.12.2019



(**Maximilian Wessendorf**)

Publikationsgenehmigung

Melder der Publikation

Hildegard Sauer

Institut für Programmstrukturen und Datenorganisation (IPD)
Lehrstuhl für Programmiersysteme
Leiter Prof. Dr. Walter F. Tichy

+49 721 608-43934
hildegard.sauer@kit.edu

Erklärung des Verfassers

Ich räume dem Karlsruher Institut für Technologie (KIT) dauerhaft ein einfaches Nutzungsrecht für die Bereitstellung einer elektronischen Fassung meiner Publikation auf dem zentralen Dokumentenserver des KIT ein.

Ich bin Inhaber aller Rechte an dem Werk; Ansprüche Dritter sind davon nicht berührt.

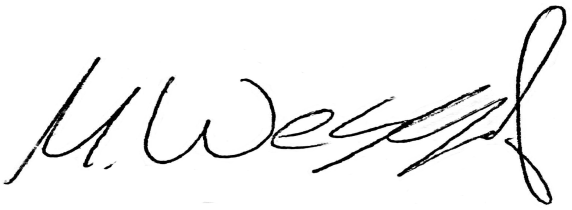
Bei etwaigen Forderungen Dritter stelle ich das KIT frei.

Eventuelle Mitautoren sind mit diesen Regelungen einverstanden.

Der Betreuer der Arbeit ist mit der Veröffentlichung einverstanden.

Art der Abschlussarbeit:	Bachelorarbeit
Titel:	Aufbau und Konsolidierung einer Konzepthierarchie für Anforderungsbeschreibungen aus unterschiedlichen Wissensquellen
Datum:	11.12.2019
Name:	Maximilian Wessendorf

Karlsruhe, 11.12.2019



(**Maximilian Wessendorf**)

Kurzfassung

Ein Problem bei der Anforderungsrückverfolgung ist, dass eine syntaktische Verbindung zwischen Begriffen in Anforderungen und Quelltext oftmals fehlt. Eine Möglichkeit Verknüpfungen dennoch korrekt herzustellen ist die Einbeziehung von Hintergrundwissen, um ein explizites Verständnis der verwendeten Begriffe zu erlangen. Eine in der Computerlinguistik bekannte Quelle für solches Hintergrundwissen über semantische Zusammenhänge ist WordNet. Um jedoch besonders für technische Begriffe eine möglichst vollständige Abdeckung zu erreichen, reicht WordNet alleine als Wissensquelle nicht aus. In dieser Arbeit wird daher ein Ansatz entwickelt um eine konsolidierte Konzepthierarchie aus mehreren beliebigen Wissensquellen aufzubauen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

In den letzten fünfzig Jahren ist Software nicht nur immer weiter in alle Bereiche unseres Lebens vorgedrungen, sondern auch immer komplexer geworden [Dvo09]. Hierdurch entstehen Bedürfnisse für neue technische Lösungen, die Softwareingenieure zum Beispiel bei der Wartung von Software oder der Auswirkungsanalyse von Anforderungsänderungen unterstützen. Die automatische Anforderungsrückverfolgung, bei der Anforderungen mit entsprechenden Quelltext-Ausschnitten, oder anderen Artefakten des Softwareentwicklungsprozesses verknüpft werden, ist ein Verfahren, das dabei helfen soll [ACC+02].
Es gibt bereits Ansätze, die derartige Verknüpfungen mit Hilfe von Informationsgewinnungstechniken (engl. information retrival) erfassen können. Diese setzen allerdings eine entsprechende Benennung der Konzepte im Anforderungsdokument und im Quelltext voraus, um eine syntaktische Verbindung herstellen zu können:

	Anforderung		**Quelltext**	
1)	... *the* $\underline{dashboard}$ ...	$\leftrightarrow$	$show(\underline{Dashboard}\ d)$ { ... }	(1.1)
2)	... *the dashboard* ...	$\times$	$show(GraphicalUserInterface\ gui)$ { ... }	

In Fall (1) könnte also durch die Wiederverwendung des Begriffes „dashboard“ eine korrekte Verbindung hergestellt werden, während in Fall (2) eine möglicherweise relevante Beziehung zwischen Anforderung und Quelltext übersehen worden wäre. In der Realität liegt eine entsprechend systematisch durchdachte Benennung möglicherweise nicht vor, was zu einer geringen Leistungsfähigkeit der Verfahren führt. Ein Grund hierfür kann sein, dass unterschiedliche Abstraktionsebenen für Konzepte in der Anforderung und dem Quelltext gewählt werden, da in der Entwurfsphase und der Implementierungsphase des Softwareentwicklungsprozesses unterschiedliche Detailgrade verwendet wurden. Daher braucht es neue Methoden, die auch ohne syntaktische Verbundenheit solche Verknüpfungen über mehrere Abstraktionsebenen hinweg herstellen können.

1.1 Anforderungsrückverfolgung mit Hintergrundwissen

Eine Möglichkeit Beziehungen zwischen syntaktisch unverbundenen Begriffen aufzudecken ist die Einbeziehung von Hintergrundwissen. Abbildung 1.1 zeigt ein Beispiel für eine Anforderung und einen Quelltext-Ausschnitt bei dem zunächst nicht ausreichend Informationen in Anforderung und Quelltext vorhanden sind, um eine Verbindung zu erfassen. Nach Hinzufügen von Weltwissen aus WordNet [Fel99] und Wikidata [VK14] bieten die hinzugefügten Konzepte, der nächst höheren Abstraktionsebene, nun die Möglichkeit eine syntaktische Verknüpfung von der Anforderung zum Quelltext herzustellen. Das Hinzufügen von

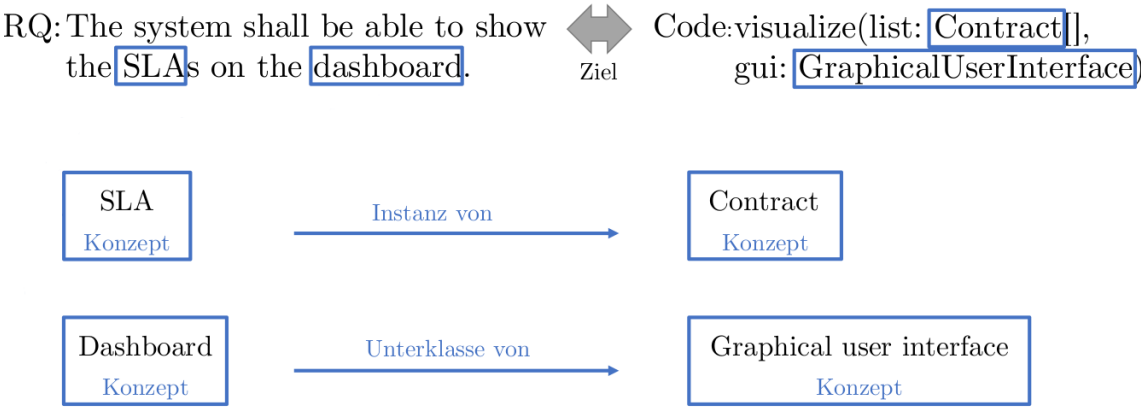


Abbildung 1.1: Beispiel für eine Anforderung und einen Quelltext-Ausschnitt, die durch Weltwissen miteinander verknüpft werden können.

Weltwissen erlaubt also, dass auch dann eine automatische Verknüpfung von Anforderungen und Quelltext erreicht werden kann, wenn in der Entwurfsphase und der Implementierungsphase des Softwareentwicklungsprozesses unterschiedliche Detailgrade verwendet werden.

1.2 Hintergrundwissen für INDIRECT

Diese Arbeit wird im Rahmen des Projektes INDIRECT des IPD Tichy verfasst. Bei INDIRECT geht es darum die Anforderungsrückverfolgbarkeit durch explizites Verständnis von Anforderungen und Quelltext wesentlich zu verbessern. Das Bereitstellen von entsprechend ausgewähltem Hintergrundwissen zur Überwindung des Eingangs beschriebenen Problems ist ein Teil dieses expliziten Verständnisses. Das Projekt wird in Kapitel 3 erklärt und diese Arbeit entsprechend eingeordnet.

1.3 Aufbau der Arbeit

In Kapitel 2 werden zunächst die zum Verständnis der Arbeit notwendigen Grundlagen aus den Bereichen Computerlinguistik, Anforderungsverwaltung und Wissensstrukturierung erläutert. In Kapitel 3 wird daraufhin das Projekt INDIRECT und seine für diese Arbeit relevanten Komponenten vorgestellt. In Kapitel 4 werden Arbeiten vorgestellt, die mit dieser Arbeit verwandt sind. Dabei werden sowohl Arbeiten betrachtet, die einen ähnlichen Ansatz verfolgen, als auch Arbeiten, die sich mit dem Aufbau und der Strukturierung von Wissensquellen unabhängig vom Themengebiet der Anforderungsverwaltung befassen. In Kapitel 5 werden die beiden Wissensquellen WordNet und Wikidata im Detail analysiert und ein Verfahren entwickelt, dass auf Basis beliebiger Wissensquellen eine Konzepthierarchie aufbauen kann. In Kapitel 6 wird beschrieben, wie der entwickelte Ansatz im Rahmen von INDIRECT als Agent implementiert wurde. In Kapitel 7 wird ausgewertet welchen Vorteil die Kombination mehrerer Wissensquellen bringen kann und wie gut das entwickelte Verfahren auf Daten aus Anforderungsdokumenten funktioniert. Abschließend wird die Arbeit in Kapitel 8 zusammengefasst und ein Ausblick gegeben.

2 Grundlagen

In diesem Kapitel werden die Grundlagen, die zum Verständnis der Arbeit nötig sind erklärt. Dazu gehören Grundlagen aus den Sprachwissenschaften, sowie der Computerlinguistik und der Anforderungsverwaltung.

2.1 Sprachwissenschaftliche Grundlagen

Im Folgenden werden zunächst grundlegende sprachwissenschaftliche Begriffe, die zum Verständnis der Arbeit nötig sind, erläutert.

2.1.1 Natürliche Sprache

Als natürliche Sprachen werden die historisch entwickelten, regional und sozial geschichteten Sprachen bezeichnet, mit denen der Mensch kommuniziert. Sie unterscheiden sich von künstlichen Sprachsystemen vor allem durch ihre lexikalische und strukturelle Mehrdeutigkeit bzw. durch ihre Vagheit und Bedeutungsvielfalt. Außerdem können sie sich über die Zeit kulturell bedingt verändern. [BGL02]

2.1.2 Entität

Eine Entität ist eine konkrete oder abstrakte Sache. In dieser Arbeit werden die in Anforderungen auftretenden Dinge als Entität bezeichnet. Sie liegen dort meist als Nominalphrase (siehe Abschnitt 2.1.7) vor.

2.1.3 Konzept

Der Begriff des Konzeptes wird, wie in Definition 2.1 beschrieben, in der Psychologie als gemeinsame Grundvorstellung definiert.

Definition 2.1: Konzept

Allgemeine Bezeichnung für eine Grundvorstellung oder eine Idee im Sinne einer Sammlung verschiedener Gedanken. [DWS17]

Das bedeutet, dass ein Konzept die gemeinsame Vorstellung verschiedener Personen von einer Sache beschreibt. Diese kann sich in verschiedenen Kulturen unterscheiden und sich

mit der Zeit verändern. Dabei können unterschiedliche Worte verwendet werden, um das gleiche Konzept zu beschreiben und gleiche Worte, um unterschiedliche Konzepte zu beschreiben. Die englischen Worte „dashboard“ und „splasher“ beschreiben beide das Konzept einer schützenden Vorrichtung, die Menschen vor Wasser oder Schlamm schützt. Das englische Wort „dashboard“ kann aber auch andere Konzepte, wie die Vorstellung einer Instrumentenanzeige in einem Fahrzeug, beschreiben.

2.1.4 Kontext

Der Kontext (Zusammenhang) beschreibt die Elemente einer Kommunikationssituation, die das Verständnis einer Äußerung bestimmen. Neben der allgemeinen Sprechsituation und persönlichen und sozialen Aspekten zählt der sprachliche Kontext (auch Kotext genannt) zu den drei Kontext-Typen. Der sprachliche Kontext verknüpft Ausdrücke grammatisch und semantisch mit ihrer Umgebung im Text. Der Kontext ist notwendig, damit aus einem Satz der aktuelle Sinn abgeleitet werden kann. So hat der Begriff „Knopf“ in einem Satz, in dem es um eine Jacke geht (sprachlicher Kontext) und der in einer Schneiderei ausgesprochen wird (allgemeine Sprechsituation) eine andere Bedeutung, als in einem Satz, der die Oberfläche einer Smartphone-Anwendung beschreibt (sprachlicher Kontext) und der in einem Anforderungsdokument vorkommt (allgemeine Sprechsituation). [BGL02]

2.1.5 Homonymie

Die Homonymie (Gleichnamigkeit) ist ein Typ lexikalischer Ambiguität. Homonyme Ausdrücke werden gleich geschrieben und ausgesprochen, obwohl sie unterschiedliche Bedeutungen haben. Ein Beispiel hierfür ist das Wort „Kiefer“, das sowohl einen Teil des menschlichen Schädels als auch eine Pflanzenart beschreibt und dabei gleich geschrieben und ausgesprochen wird. Welche Bedeutung ein homonymer Ausdruck in einem Satz hat, hängt vom Kontext ab (siehe Abschnitt 2.1.4). [BGL02]

2.1.6 Semantische Relation

Der Begriff der semantischen Relation umfasst alle Relationen, die zwischen den Bedeutungen von natürlichsprachlichen Ausdrücken (Wörtern, Sätzen) bestehen. Im Folgenden sind die wichtigsten semantischen Relationen aus Bußmanns Lexikon der Sprachwissenschaften [BGL02] aufgelistet:

Hyperonymie umfasst semantische Relationen der lexikalischen Überordnung zur Kennzeichnung hierarchieähnlicher Gliederungen des Wortschatzes. Frucht ist zum Beispiel ein Hyperonym von Apfel, Birne und Pflaume. Der Übergang von Apfel zu Frucht bringt eine Verallgemeinerung der Bedeutung mit sich.

Hyponymie beschreibt die semantische Relation der Unterordnung im Sinne einer inhaltsmäßigen Spezifizierung. So ist zum Beispiel Apfel ein Hyponym zu Frucht, da Apfel eine spezifischere Bedeutung als Frucht hat. Jedes Hyponym unterscheidet sich von seinen Hyperonymen durch mindestens ein weiteres spezifizierendes Merkmal.

Eine **Teil-von-Relation** ist eine Semantische Relation zwischen sprachlichen Ausdrücken zur Bezeichnung der Beziehung eines Teils zum Ganzen bzw. zur Bezeichnung von Besitzverhältnissen. Ein Haus hat zum Beispiel eine Tür und eine Tür ist Teil von einem Haus. Der Begriff Haus wird als **Holonym** des Begriffs Tür bezeichnet und Tür als **Meronym** von Haus.

Synonymie ist die semantische Relation der Bedeutungsgleichheit bzw. Bedeutungsähnlichkeit von zwei oder mehreren Ausdrücken. Es wird zwischen totaler und partieller Synonymie unterschieden. Bei der totalen Synonymie muss ein Ausdruck in jedem erdenklichen

Kontext durch den anderen ersetzbar sein. Ein Beispiel für zwei Synonyme Begriffe sind „Vorschlag“ und „Anregung“.

Antonymie ist ein Oberbegriff für semantische Gegensatzrelationen. Zwei Ausdrücke A und B stehen in einer solchen Gegensatzrelation wenn gilt: Trifft A zu, kann B nicht zutreffen und umgekehrt. Ein Beispiel für zwei antonyme Begriffe sind „Verlust“ und „Gewinn“.

2.1.7 Phrase

Phrase ist die Bezeichnung für eine syntaktisch zusammengehörige Wortgruppe ohne finite Verbform. Eine der wichtigsten Arten der Phrase ist die Nominalphrase. Sie besteht aus einem auf Nomen basierenden Ausdruck mit entsprechenden attributiven Erweiterungen. Beispiele für Nominalphrasen sind die Namen sämtlicher Entitäten, für die im Rahmen der Arbeit eine Konzepthierarchie aufgebaut wird. [BGL02]

2.1.8 Lexikalischer Kopf

Der lexikalische Kopf ist der Teil eines zusammengesetzten Ausdrucks, der den gleichen Kategorietypen aber eine niedrigere Komplexität hat als der Ausdruck selbst. So ist für den Ausdruck „setting of system“ der Begriff „setting“ der lexikalische Kopf, da der Ausdruck eine Einstellung (engl. setting) mit der Spezifizierung, dass es sich um eine Einstellung für ein System (engl. system) handelt, beschreibt. [BGL02]

2.2 Grundbegriffe des Requirements Engineering (RE)

Im Folgenden werden grundlegende Begriffe aus dem Requirements Engineering, die zum Verständnis der Arbeit nötig sind, erläutert.

2.2.1 Anforderung

Eine Anforderung (engl. requirement) beschreibt, wie in Definition 2.2 erklärt, einen Bedarf und die mit ihm verbundenen Einschränkungen und Bedingungen.

Definition 2.2: Anforderung

A requirement is a statement which translates or expresses a need and its associated constraints and conditions. [ISO]

Anforderungen werden standardmäßig in natürlicher Sprache verfasst und neben zusätzlichen Informationen wie einem Glossar und erklärenden Diagrammen in einer Software Requirements Specification (SRS) aufgelistet. „KeePass can support importing data from CSV files, Code Wallet, Password Safe and Personal Vault.“ ist ein Beispiel für eine Anforderung, die eine Funktion des Systems beschreibt.

2.2.2 Requirements Engineering

Die Disziplin des Requirements Engineering vermittelt, wie in Definition 2.3 erklärt, zwischen Abnehmer und Lieferant mit dem Ziel die vereinbarten Anforderungen (siehe Abschnitt 2.2.1) für ein System, eine Software oder einen Service zunächst aufzustellen und dann zu verwalten.

Definition 2.3: Requirements Engineering

Requirements engineering is the interdisciplinary function that mediates between the domains of the acquirer and supplier to establish and maintain the requirements to be met by the system, software or service of interest. [ISO]

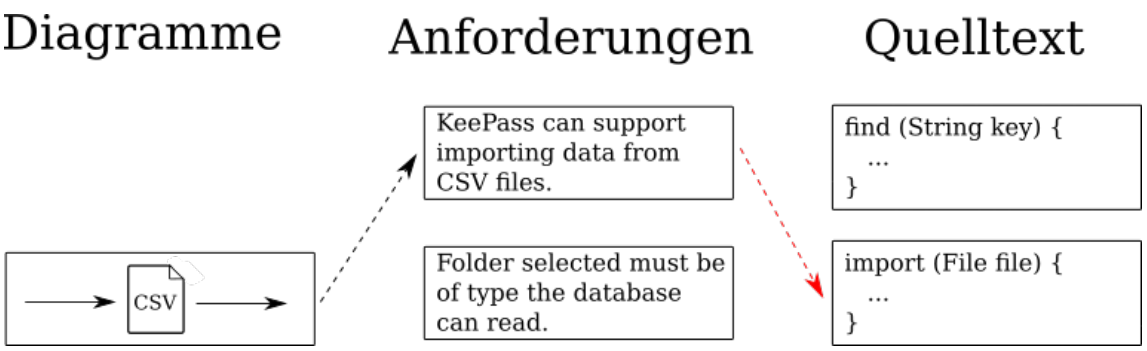


Abbildung 2.1: Die Anforderungsrückverfolgbarkeit beschäftigt sich mit der Identifikation und Dokumentation, der Beziehungen einer Anforderungen zu den anderen Anforderungen und umliegenden Elementen.

2.2.3 Anforderungsrückverfolgbarkeit

Die Disziplin der **Anforderungsrückverfolgbarkeit** (engl. Requirements tracing) beschäftigt sich mit der Identifizierung und Dokumentation des Ableitungspfades und des Zuleitungspfades von Anforderungen [ISO]. Der Zuleitungspfad umfasst die Elemente, aus denen eine Anforderung abgeleitet wird. Dazu können zum Beispiel erste Funktionsbeschreibungen oder Diagramme gehören, die in der Anforderung spezifiziert werden. Der Ableitungspfad umfasst die Elemente, die aus einer Anforderung abgeleitet werden. Dazu gehören zum Beispiel der Quelltext, der eine Anforderung implementiert oder Testfälle, die die korrekte Implementierung der Anforderung prüfen. Ein Beispiel für eine Anforderung und ihren Zu- und Ableitungspfad ist in Abbildung 2.1 dargestellt.

2.3 Computerlinguistik

Im Folgenden werden grundlegende Begriffe der Computerlinguistik (engl. Natural Language Processing), die zum Verständnis der Arbeit nötig sind, erläutert.

2.3.1 Disambiguierung

Disambiguierung beschreibt den Vorgang der Auflösung lexikalischer oder struktureller Mehrdeutigkeit durch den Kontext [BGL02]. Ein Beispiel für eine Mehrdeutigkeit ist die Homonymie (siehe Abschnitt 2.1.5). Ein Beispiel für Homonymie ist der Begriff „Kiefer“, der sowohl einen Teil des menschlichen Schädels als auch eine Pflanzenart beschreibt. Disambiguierung ist der Vorgang der aufdeckt, dass es sich in dem Satz „Er ging in den Wald und fällte eine Kiefer.“ um die Kiefer im Sinne einer Pflanzenart handelt. Diese Tatsache ergibt sich dabei aus dem Kontext. So ist der Kiefer im Sinne eines Teils des Schädels unverträglich mit der Aussage „ jemand fällt X“ und der Begriff „Wald“, der im gleichen Satz vorkommt, ist ein zusätzlicher Indikator.

2.3.2 Bag-of-Words Modell

Das Bag-of-Words Modell ist eine Möglichkeit Text zu analysieren. Es werden Worte als Menge mit entsprechender Anzahl des Auftretens in einem Dokument dargestellt. Die Reihenfolge der Worte oder ihre syntaktische Struktur wird dabei ignoriert. Satzzeichen, sowie Worte, die nicht zum Kontext beitragen, werden meist aus der Menge entfernt. Außerdem werden alle Worte auf ihren Wortstamm reduziert. [MCG16]

2.3.3 Tf-idf-Maß

Das Tf-idf-Maß (Abkürzung für engl. term frequency - inverse document frequency) ist ein Maß zur Gewichtung von Begriffen in Dokumenten. Der Wert zur Gewichtung errechnet sich als Produkt der inversen Auftrittshäufigkeit (IDF) eines Begriffes in Dokumenten und der Auftrittshäufigkeit dieses Begriffes (TF) in einem betrachteten Dokument. Erstere wird als Quotient aus der Anzahl der Dokumente und der Anzahl der Dokumente in denen der betrachtete Begriff vorkommt berechnet. Die IDF ist also maximal, wenn ein Begriff nur im betrachteten Dokument vorkommt und minimal, wenn er in allen Dokumenten vorkommt. Die TF für einen Begriff i in einem Dokument j stellt die Auftrittshäufigkeit von i in j dar. Das Tf-idf-Maß bevorzugt also Begriffe, die im betrachteten Dokument möglichst häufig und gleichzeitig selten in allen anderen Dokumenten sind. Die Idee hinter dieser Gewichtung besteht darin, dass ein Begriff der nur in einem bestimmten Dokument verwendet wird eine höhere Aussagekraft über den Inhalt des Dokumentes haben soll als ein Begriff der über alle Dokumente hinweg immer wieder auftritt. [JM09]

2.4 Repräsentationen von Wissen

Im Rahmen dieser Arbeit spielt die Repräsentation von Wissen immer wieder eine wichtige Rolle. Zum einen wird aus verschiedenen Quellen Hintergrundwissen extrahiert, um die Entitäten in Anforderungsdokumenten einzuordnen. Zum anderen stellen die Ergebnisse dieser Extraktion und Konsolidierung selbst wieder eine Art von Wissensquelle dar. Im Folgenden werden grundlegende Modelle zur Repräsentation von Wissen, die zum Verständnis der Arbeit nötig sind, erläutert.

2.4.1 Konzepthierarchie

Im Rahmen dieser Arbeit wird immer wieder der Begriff der Konzepthierarchie verwendet. Wie in Definition 2.4 beschrieben, handelt es sich dabei um eine Menge von Konzepten C und eine Halbordnung (partielle Ordnung) $\leq$ auf diesen.

Definition 2.4: Konzepthierarchie

Unter einer Konzepthierarchie (C, $\leq$) versteht man eine nichtleere endliche Menge C, deren Elemente Konzepte oder Begriffe genannt werden, und eine partielle Ordnung $\leq$ auf C, so dass in C ein bezüglich $\leq$ größtes Element $\top$ und ein bezüglich $\leq$ kleinstes Element $\bot$ existiert. Für c, c' $\in$ C mit c $\leq$ c' nennt man c einen Unterbegriff von c' bzw. c' einen Oberbegriff von c. [Udo06]

Die Halbordnung $\leq$ beschreibt die Beziehung zweier Konzepte $c \leq c'$, wobei hier c ein Unterkonzept von c' und umgekehrt c' ein Überkonzept von c ist. Eine Halbordnung ist eine reflexive, antisymmetrische und transitive Relation. Das bedeutet, dass ein Konzept x sowohl Überkonzept als auch Unterkonzept von sich selbst ist (Reflexivität), dass ein Konzept x nicht Unterkonzept eines Konzeptes y sein kann, wenn x bereits Überkonzept von y ist (Antisymmetrie) und dass ein Konzept x, das ein Unterkonzept von y ist, welches wiederum ein Unterkonzept von z ist, selbst auch ein Unterkonzept von z ist (Transitivität).

2.4.2 Ontologie

Ein weiterer Begriff, der im Rahmen dieser Arbeit immer wieder im Zusammenhang mit Wissensquellen verwendet wird, ist der in Definition 2.5 definierte Begriff der Ontologie.

Definition 2.5: Ontologie

A logical structure of the terms used to describe a domain of knowledge, including both the definitions of the applicable terms and their relationships. [IEE]

Ontologien beschreiben also Wissen durch eine Struktur aus Termen, deren Erklärung und deren Beziehungen untereinander. Der Begriff der Ontologie beschreibt grundsätzlich die gleiche Art von Struktur wie eine Konzepthierarchie (siehe Abschnitt 2.4.1), wobei sich der Begriff dabei weniger auf die konkrete Art der Struktur, sondern eher auf deren Inhalt bezieht.

Einen Überblick über Ontologien gibt Roche in [Roc03]. Demnach können Ontologien für verschiedene Arten von Wissen entwickelt werden. Ein Beispiel hierfür ist die generische Ontologie (engl. generic ontology oder top ontology). Sie enthält generelles Wissen, unabhängig von bestimmten Anwendungsfällen und kann daher für verschiedenste Aufgaben wiederverwendet werden. Ein anderes Beispiel sind Domänen-Ontologien (engl. domain ontology). Sie enthalten Wissen über eine bestimmte Domäne und können daher nur für Aufgaben innerhalb dieser wiederverwendet werden.

In einigen Anwendungsfällen kann es hilfreich sein mehrere Ontologien einzubeziehen. Dafür haben sich verschiedene Begriffe gebildet, die in der Literatur teilweise nicht eindeutig voneinander abgegrenzt sind. Im Folgenden wird ein Verständnis dieser Begriffe basierend auf [Kle01] vorgestellt. Die Verwendung der Begriffe in dieser Arbeit orientiert sich an diesen.

Das **Übersetzen (translating)** einer Ontologie beschreibt das Verändern der verwendeten Ontologiesprache ohne den Inhalt zu verändern. Möchte man mit mehreren Ontologien, die in verschiedenen Ontologiesprachen dargestellt sind arbeiten, so muss man zunächst alle in eine einheitliche Darstellungsform übersetzen.

Das **Kombinieren (combining)** mehrerer Ontologien beschreibt allgemein das Verwenden mehrerer Ontologien zur Erfüllung einer Aufgabe, für die deren gegenseitige Beziehung relevant ist. Dies kann besonders dann sinnvoll sein, wenn zwei verschiedene Ontologie sich gegenseitig ergänzende Informationen enthalten.

Das **Abbilden (mapping)** mehrerer Ontologien beschreibt das Bestimmen von Äquivalenzbeziehungen für Konzepte und Relationen der verschiedenen Ontologien. Bevor Ontologien kombiniert werden können, ist es zunächst notwendig solche Äquivalenzbeziehungen zu bestimmen. Ansonsten ist nicht klar, welche Konzepte einer Ontologie den Konzepten einer anderen Ontologie entsprechen und für welche Konzepte es kein entsprechendes Konzept in der anderen Ontologie gibt. Außerdem kann nur mit den Relationen der verschiedenen Ontologien gearbeitet werden, wenn abgebildet wurde, welche davon sich entsprechen, widersprechen oder ergänzen.

Das **Zusammenführen (merging, integration)** mehrerer Ontologien beschreibt die Erstellung einer neuen Ontologie aus zwei oder mehr, sich gegenseitig überschneidenden Ontologien. Dazu werden die Informationen aus der gegenseitigen Abbildung der Ontologien aufeinander verwendet. Sind Konzepte oder Relationen in den Quellontologien zum Beispiel identisch, ergibt sich in der Zielontologie nur noch ein entsprechendes Konzept bzw. eine Relation.

3 INDIRECT: INtent-DrIven REquirements-to-Code Traceability

Diese Arbeit ist Teil des Projektes INDIRECT. Dieses zielt darauf ab, die Präzision der automatischen Rückverfolgung von Anforderungen, also der Abbildung von Anforderungen auf die aus ihnen resultierenden Stellen im Quelltext signifikant zu erhöhen. Dafür wird ein neuer Ansatz vorgeschlagen, bei dem explizite Absichtsmodelle für die in natürlicher Sprache verfassten Anforderungen und für den Quelltext erzeugt werden. Absichtsmodelle sind Graphrepräsentationen, die die Bedeutung von Anforderung und Quelltext darstellen. Die Verknüpfung der Absichtsmodelle von Anforderung und den entsprechenden Teilen des Quelltextes wird durch übereinstimmende Muster in den Graphstrukturen der beiden Modelle gewonnen.

3.1 Projektarchitektur

INDIRECT verwendet die agentenbasierte Architektur PARSE (ehemals ProNat) [WT15] (siehe Abbildung 3.1). Dabei dient ein zentraler Datenspeicher als Schnittstelle für verschiedene Agenten, die die Daten kontinuierlich ergänzen. Neue Eingaben werden hierbei zunächst mit seichter Sprachverarbeitung vorbereitet. Die Anforderungssätze werden dabei in die einzelnen Wörter (Token) zerteilt und diese um erste Zusatzinformationen, wie ihre Wortart, ergänzt. Diese Token werden dann in einen Graphen überführt. Die verschiedenen Agenten ergänzen dann diesen Graphen entsprechend ihrer Aufgabe solange bis dieser sich nicht mehr verändert.

3.2 Extraktion von Entitäten

Einer der bereits existierenden Agenten in INDIRECT ist der EntityRecognizer. Seine Aufgabe ist es Entitäten aus den untersuchten Sätzen zu extrahieren. Dazu verwendet er den Stanford Parser um Nomen zu erkennen und aus den syntaktischen Beziehungen rund um die erkannten Nomen weitere Worte zu finden, die sich auf die durch das Nomen beschriebene Entität beziehen. Die Ergebnisse des Entität-Erkenners sind die Grundlage zur in dieser Arbeit behandelten Konzeptualisierung.

3.3 Konzeptualisierung

Ziel der Arbeit ist, wie in Kapitel 1 eröffnet, die Anreicherung der in den untersuchten Anforderungen auftretenden Entitäten mit Hintergrundwissen. Es wurde bereits ein Agent

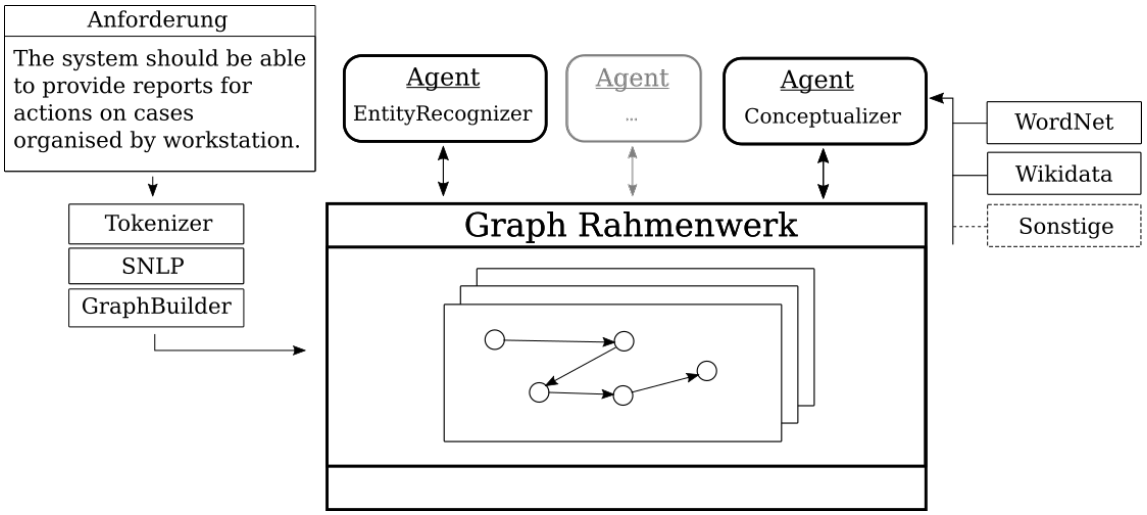


Abbildung 3.1: Vereinfachte Darstellung der Projektarchitektur von INDIRECT basierend auf PARSE.

erstellt, der die gefundenen Entitäten mit Hintergrundwissen aus WordNet anreichern soll. Allerdings trat dabei das Problem auf, dass WordNet als einzelne Wissensquelle viele in Anforderungen auftretenden technischen Begriffe nicht kennt. In dieser Arbeit wird der als Konzeptualisierer bezeichnete Agent deshalb grundlegend überarbeitet, sodass er Wissen aus verschiedensten Quellen anreichern und konsolidieren kann. Die dabei gewonnen Informationen werden dann in die Graphrepräsentation hinzugefügt.

4 Verwandte Arbeiten

Ziel dieser Arbeit ist der Aufbau und die Konsolidierung einer Konzepthierarchie zur Verbesserung der Rückverfolgungsergebnisse im Rahmen von INDIRECT (siehe Kapitel 3). Die Einbeziehung von Hintergrundwissen wurde bereits in anderen Arbeiten, die sich mit der Thematik der Rückverfolgung von Anforderungen beschäftigen, erprobt. In **??** wird zunächst eine Arbeit betrachtet, die Hintergrundwissen als Kern der Berechnung der Abbildungen bei der Rückverfolgung von Anforderungen verwendet. Bei dieser und weiteren Arbeiten [HYS10], sowie der ersten Version des Konzeptualisierers im Rahmen von INDIRECT, tritt unter anderem der Mangel an Wissensquellen mit entsprechenden technischen Begriffen als zentrales Problem auf. In Abschnitt **??** werden deshalb, unabhängig vom Anwendungsfall der Anforderungsrückverfolgung, Arbeiten betrachtet, die sich mit dem Verbinden und Anreichern von Wissensquellen beschäftigen. Dabei liegt der Fokus auf der Einbeziehung beliebig vieler, unabhängiger und komplementärer Wissensquellen. Im Folgenden tritt der Begriff der Ontologie (siege Kapitel2) als ein in der wissenschaftlichen Literatur gängiger Begriff für eine Art der Repräsentation von Hintergrundwissen auf.

4.1 Einsatz von Hintergrundwissen bei der Rückverfolgung von Anforderungen

In ihrer Arbeit **Ontology-based trace retrieval** [LCH13] verwenden Li und Cleland-Huang Hintergrundwissen in Form von Ontologien um die Rückverfolgbarkeit von Anforderungen zu verbessern. Dazu haben sie ein Verfahren entwickelt, welches alle Quell- und Zielartefakte miteinander vergleicht und anhand deren Beziehungen in einer Ontologie einen Wert für deren Ähnlichkeit berechnet. Aus einem Quell- und Zielartefakt in Form von natürlichsprachlichen Sätzen werden zunächst mit Hilfe des Stanford Parsers Phrasen extrahiert. Diese Phrasen werden dann mit dem Tf-idf-Maß (siehe Kapitel 2) gewichtet, sodass selteneren Phrasen eine höhere Bedeutung zugemessen wird als häufigeren. Dazu wird das Ergebnis der Division der Häufigkeit einer Phrase in einem Artefakt N(p) durch die Häufigkeit derselben Phrase in allen Artefakten QF(p) als Gewichtungsfaktor W(p) gewählt. Anschließend wird für die Phrasen in Quell- und Zielartefakt jeweils paarweise die Ähnlichkeit berechnet. Dazu werden die Phrasen in Terme aufgeteilt. Diese Terme können Nomen, Verben oder Teilphrasen sein. Ist die gesamte Phrase als ein Element bereits in der Ontologie enthalten, so wird sie als ein primitiver Term betrachtet. Falls nicht wird die Phrase in einzelne Terme aufgeteilt und diese wiederum in der Ontologie gesucht. So wird

„UVHS message“ als Term betrachtet, falls das Konzept „UVHS message“ in der Ontologie existiert. Existiert das Konzept nicht, werden „UVHS“ und „message“ als einzelne Terme betrachtet. Die gewonnen Terme werden ebenfalls wieder mit dem Tf-idf-Maß gewichtet. Die Ähnlichkeit zweier Termen wird mit folgender Formel berechnet:

$$sim(t_1, t_2) = \begin{cases} 1 & t_1 = t_2 \\ 0 & t_1 \, oder \, t_2 \notin Ontologie \\ 1/shortestPath(t_1, t_2) & sonst \end{cases} \tag{4.1}$$

Das Maß der Ähnlichkeit zweier Terme beträgt also eins wenn sie identisch sind und null wenn einer der Terme nicht in der Ontologie enthalten ist. Sind beide Terme in der Ontologie enthalten und nicht gleich, wird ihre Ähnlichkeit über die Länge des kürzesten Pfades zwischen ihnen in der Ontologie berechnet. Basierend auf der Term-zu-Term Ähnlichkeit kann dann die Term-zu-Phrase Ähnlichkeit mit folgender Formel berechnet werden:

$$sim(t, P < t_1, t_2, ..., t_i >) = max(sim(t, t_i)) \tag{4.2}$$

Das Maß der Ähnlichkeit zwischen einem Term und einer Phrase berechnet sich also als maximale Ähnlichkeit zwischen dem Term und einem beliebigen Term in der Phrase. Neben der Gewichtung und der Distanz spielt auch die Reihenfolge der Worte einer Phrase eine Rolle. Die Phrasen „yellow light“ und „light yellow“ beziehen sich auf zwei verschiedene Konzepte. Bei der ersten Phrase geht es um Licht, während es bei der zweiten Phrase um eine Farbe handelt. Deshalb wird der Kopf (siehe Kapitel 2) der Phrase als Schlüsselterm in das Verfahren einbezogen. Der Kopf der Phrase ist gewöhnlich der letzte Term der Phrase. Ausnahmen sind jedoch möglich. So identifizieren Li und Cleland-Huang die Fälle (1) {[noun, verb]}<noun><prep><noun> und (2) {[noun, verb]}<noun><attributive clause> als Ausnahmen. Im ersten Fall ist das Nomen vor der Präposition der Schlüsselterm. Im zweiten Fall ist das Nomen vor der attributiven Klausel der Schlüsselterm. Ein Beispiel für den ersten Fall stellt die Phrase „message of priority“ dar, bei der offensichtlich das Konzept der Nachricht im Zentrum der Aussage steht und „of priority“ einen Zusatz darstellt. Ein Beispiel für den zweiten Fall stellt die Phrase „message that has priority“ dar, bei dem ebenfalls das Konzept der Nachricht im Zentrum steht und eine Spezialisierung angehängt wird. Der Standardfall wäre die Phrase „priority message“, bei dem das letzte Nomen der Schlüsselterm ist. Auf Basis dieser beiden Ähnlichkeitsmaße, den Gewichten, sowie der Beachtung des Schlüsselterms einer Phrase ergibt sich folgende Formel als Maß der Ähnlichkeit zwischen zwei Phrasen:

$$\begin{aligned} sim(P_1, P_2) = & \frac{\sum(w(t_i) \cdot sim(t_i, P_2))}{\sum w(t_i)} \cdot max(w(t_i)) \cdot \frac{max(w(t_i))}{max(w(t_i)) + w(t_{key})} \\ & + sim(t_{key}, P_2) \cdot \frac{w(t_{key})}{max(w(t_i)) + w(t_{key})} \end{aligned} \tag{4.3}$$

Dabei beschreibt t_i den i-ten Term in der Phrase P_1. Der erste Faktor beschreibt die gewichtete durchschnittliche Ähnlichkeit der Terme in P_1 gegenüber der Phrase P_2. Der zweite Faktor beschreibt das maximale Gewicht eines Terms aus P_1. Der dritte Faktor beschreibt den Beitrag zwischen gewöhnlichen Termen und dem Schlüsselterm. Auf dieses Produkt wird dann die Ähnlichkeit zwischen dem Schlüsselterm aus P_1 und der Phrase P_2 addiert und diese wiederum mit dem Beitrag zwischen Schlüsselterm und gewöhnlichen Termen in umgekehrter Gewichtung multipliziert. Ist also das Gewicht des Schlüsselterms groß, so trägt vor allem seine Ähnlichkeit bei, während bei geringerer Gewichtung des

Tabelle 4.1: Evaluation des Verfahrens von Li und Cleland-Huang

Method	MAP of 40 source artifacts	MAP of entire dataset
VSM	0.116	0.257
Generelle Ontologie	0.162	0.262
Beide Ontologien	0.386	-

Schlüsselterms die anderen Terme stärker beitragen. Der gleiche Ansatz zur Bestimmung der Ähnlichkeit wird nun auf der Ebene der Phrasen und der Sätze, also der Ausgangsartefakte, wiederholt. Dabei beschreibt folgende Formel das Maß der Ähnlichkeit zwischen einer Phrase und einem Satz:

$$sim(p, S < p_1, p_2, ..., p_i >) = max(sim(p, p_i)) \tag{4.4}$$

Daraus ergibt sich dann das Maß für die Ähnlichkeit zweier Sätze und damit dem Quell- und Zielartefakt:

$$sim(S_1, S_2) = \frac{\sum w(p_i) \cdot sim(p_i, S_2))}{\sum w(p_i)} \cdot max(w(p_i)) \tag{4.5}$$

Im Grunde nutzt das von Li und Cleland-Huang entwickelte Verfahren Hintergrundwissen um die Ähnlichkeit zweier Terme zu berechnen und schließt dann unter Einbeziehung entsprechender Gewichtungen und der Rolle der Terme im Satz auf die Ähnlichkeit von ganzen Sätzen. Den Ansatz evaluieren sie mit zwei verschiedenen Ontologien. Zum einen wurde eine Domänen-Ontologie konstruiert und zum anderen eine generelle Ontologie. Während die generelle Ontologie aus vorhandenen Wissensquellen wie WordNet konstruiert werden kann, wird die Domänen-Ontologie manuell, auf den Testdatensatz zugeschnitten, erstellt. Dieses Vorgehen nennen die Autoren als hauptsächliche Einschränkung der Validität des Ergebnisses. Der Datensatz zur Evaluation besteht aus 226 Anforderungen als Quellartefakte und 1591 Designelementen als Zielartefakte. Beide liegen als natürlichsprachliche Sätze vor. Messwert zur Evaluation ist die mittlere durchschnittliche Präzision (engl. Mean Average Precision, MAP), also das Mittel der durchschnittlichen Präzisionen, die für jedes Quellartefakt berechnet werden. Diese wird jeweils für die generelle Ontologie alleine und die Kombination aus genereller und Domänen-Ontologie berechnet. Als Grundlage zur Messung der Leistung des Verfahrens dient der Vergleich mit dem Vektorraum-Modell (engl. Vector Space Model), einem Verfahren das bisher zur Rückverfolgung von Anforderungen zum Einsatz kommt. In Tabelle 4.1 ist die MAP der drei Versuche einmal für vierzig ausgewählte Quellartefakte und einmal für den gesamten Datensatz aufgeführt. Die Verwendung der generellen Ontologie hat dabei eine geringfügig höhere MAP als bei der Verwendung des Vektorraum-Modells. Bei der Verwendung der Kombination aus Domänen und genereller Ontologie, die aufgrund der limitierten Größe der Domänen-Ontologie nur auf vierzig Artefakten ausgeführt werden kann, wird mit einer MAP von 0.386 ein deutlicher höherer Wert als bei der Verwendung der generellen Ontologie (0.162) und Vektorraum-Modells (0.116) erzielt. Li und Cleland-Huang zeigen mit ihrem Ansatz die Einsatzmöglichkeit von Hintergrundwissen in der Anforderungsverfolgung unter der Prämisse, dass entsprechende Wissensquellen vorhanden oder praktikabel konstruierbar sind.

4.2 Anreicherung von Hintergrundwissen zur Anforderungsanalyse

Wie bereits zu Beginn des Kapitels erwähnt wurde mangelt es oft an entsprechenden Wissensquellen. Die meisten Arbeiten umgehen das Problem in dem sie Ontologien manuell

Algorithm 1 The mapping algorithm.

Input: $Senses_{\text{Wiki}}$, $Senses_{\text{WN}}$
Output: a mapping $\mu : Senses_{\text{Wiki}} \rightarrow Senses_{\text{WN}} \cup \{\epsilon\}$

1: **for each** $w \in Senses_{\text{Wiki}}$
2: $\quad \mu(w) := \epsilon$
3: **for each** $w \in Senses_{\text{Wiki}}$
4: $\quad$ **if** $|Senses_{\text{Wiki}}(w)| = |Senses_{\text{WN}}(w)| = 1$ **then**
5: $\quad\quad \mu(w) := w_n^1$
6: **for each** $w \in Senses_{\text{Wiki}}$
7: $\quad$ **if** $\mu(w) = \epsilon$ **then**
8: $\quad\quad$ **for each** $d \in Senses_{\text{Wiki}}$ s.t. d redirects to w
9: $\quad\quad\quad$ **if** $\mu(d) \neq \epsilon$ and $\mu(d)$ is in a synset of w **then**
10: $\quad\quad\quad\quad \mu(w) :=$ sense of w in synset of $\mu(d)$; **break**
11: **for each** $w \in Senses_{\text{Wiki}}$
12: $\quad$ **if** $\mu(w) = \epsilon$ **then**
13: $\quad\quad$ **if** no tie occurs **then**
14: $\quad\quad\quad \mu(w) := \underset{s \in Senses_{\text{WN}}(w)}{\text{argmax}} \; p(s|w)$
15: **return** μ

Abbildung 4.1: Algorithmus zur Erstellung einer Abbildung von Konzepten aus Wikipedia auf Konzepte aus WordNet oder das leere Wort.

und spezifisch für ihre Evaluation erstellen. Dies sorgt jedoch dafür, dass Ergebnisse der Arbeiten möglicherweise verfälscht werden und die vorgeschlagenen Methoden in der Praxis nicht automatisiert einsetzbar sind. Im Folgenden werden daher Arbeiten untersucht, die Ansätze liefern, mit denen möglichst vollständiges Hintergrundwissen aus mehreren Quellen erlangt werden kann. Dafür werden zum einen Arbeiten untersucht, die möglichst vollständiges Wissen durch die Kombination mehrerer komplementärer und strukturierter Wissensquellen erlangen und zum anderen solche, die bestehende strukturierte Wissensquellen mit Wissen aus natürlicher Sprache ergänzen.

4.2.1 Anreicherung von Hintergrundwissen aus strukturierten Wissensquellen

Wissensquellen in strukturierter Form werden klassischerweise als Graph modelliert. Knoten stellen dabei Konzepte dar und Kanten die semantischen Relationen zwischen diesen. Beispiele für existierende Wissensquellen sind hierbei WordNet [Fel99] und Wikidata [VK14]. Die verschiedenen Wissensquellen sind auf unterschiedliche Weise konstruiert worden und sind deshalb in einigen Bereichen komplementär. Um diesen Effekt zu nutzen muss Wissen aus verschiedenen Quellen zusammengeführt werden.

Ein praktisches Beispiel für die automatische Kombination von Wissensquellen stellt die Arbeit **BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network** [NP12] dar. Ziel ist es das lexikographische Wissen aus WordNet und das enzyklopädische Wissen aus Wikipedia zu verbinden. Dazu wurde ein Verfahren entwickelt, das eine automatische Abbildung

$$\mu : Senses_{\text{Wiki}} \rightarrow Senses_{\text{WN}} \cup \varepsilon \tag{4.6}$$

mit einem F_1-Wert von bis zu 78% herstellen konnten.

In Abbildung 4.1 ist der entwickelte Abbildungsalgorithmus dargestellt. Zunächst ist die Abbildung μ leer, das heißt sie bildet von jeder Wikipediaseite auf das leere Wort ab (Zeilen 1-2). Als erstes werden alle Wikipediaseiten deren Titel sowohl in Wikipedia als auch in WordNet monosem sind auf deren einzige Bedeutung in WordNet abgebildet (Zeilen 3-5). Für jede weitere Wikipediaseite w, die bis jetzt noch auf das leere Wort abbildet, wird für jede Weiterleitungsseite d auf w in Wikipedia untersucht, ob diese bereits abgebildet wurde. Ist dies der Fall wird, falls vorhanden, w auf einen Begriff in WordNet abgebildet, der in

einem gemeinsamen Synonymring mit einem Begriff liegt auf den d bereits abbildet (Zeilen 8-10). Die Betrachtung von Synonymen schränkt also die Auswahl bei der Abbildung, wenn Begriffe nicht monosem sind, auf einen Begriff ein. Letztlich werden alle Wikipediaseiten w, die bis jetzt noch nicht abgebildet wurden, auf das am wahrscheinlichsten passende Konzept s in WordNet abgebildet (Zeilen 11-14). Die Wahrscheinlichkeit für eine Abbildung von einer Wikipediaseite auf ein Konzept aus WordNet wird auf zwei Arten berechnet und mit der Summe aller Werte gewichtet.

Bei der ersten Variante wird der Wert auf Basis des Bag-of-words Modells berechnet. Der Wert ergibt sich dabei als

$$score(s, w) = |Ctx(s) \cap Ctx(w)| + 1 \tag{4.7}$$

,wobei Ctx(s) und Ctx(w) jeweils die Disambiguierungskontexte in Form von Mengen von Worten sind und eins als Glättungsfaktor hinzu addiert wird.

Bei der zweiten Variante wird der Wert auf Basis der Distanz von Worten im WordNet Graphen bestimmt. Dazu werden alle WordNet Bedeutungen für einen Wikipediaseitentitel und alle WordNet Bedeutungen für die Worte in dessen Disambiguierungskontext als Knoten zu einem leeren Graphen hinzugefügt. Der Graph wird dann um Kanten ergänzt, die einen Pfad mit maximaler Länge L zwischen den Knoten in WordNet repräsentieren. Der Wert wird dann als

$$score(s, w) = \sum_{cw \in Ctx(w)} \sum_{s' \in SensesWN(cw)} \sum_{p \in pathsWN(s,s')} e^{-(length(p)-1)} \tag{4.8}$$

also der Summe aller minimalen Pfadlängen zwischen den WordNet Konzepten der um w liegenden Konzepte zum untersuchten Konzept s. Die Pfadlängen werden als negativer Exponent der e-Funktion addiert, sodass ein kurzer Pfad exponentiell besser als ein längerer Pfad ist.

Zur Evaluation des Mapping-Verfahrens wurde manuell ein Datensatz erstellt der eintausend Abbildungen von Wikipediaseiten auf das entsprechende Konzept in WordNet enthält. Die Abbildungen wurden dann mit der BoW und der Graph Methode berechnet und jeweils nach Präzision, Ausbeute und F_1 ausgewertet. Außerdem wurden für die Disambiguierungskontexte verschiedene Umfänge ausgewertet. Dabei wurden jeweils nur die Konzepte in der Umgebung, die Glosse, oder beides verwendet. Bei der Graphmethode wurden außerdem verschiedene maximale Tiefen für die Pfade evaluiert. Der beste F_1-Wert von 77.7 wurde mit der Graph-Methode, unter Einbeziehung von Umgebungskonzepten und Glosse, bei einer maximalen Pfadtiefe von zwei erreicht. Der drittbeste F_1-Wert von 75.0 wurde mit der BoW-Methode, unter Einbeziehung von Umgebungskonzepten und Glosse erreicht. Als Vergleich wurde ein F_1-Wert von 31.9 bei zufälliger Auswahl des Zielkonzeptes ermittelt. Mit der Arbeit geben die Autoren einen Überblick über den Stand der Technik hinsichtlich der Abbildbarkeit der Konzepte von zwei bekannten Wissensquellen.

In seiner Arbeit **Combining and relating ontologies: an analysis of problems and solutions** [Kle01] analysiert Klein die grundlegenden Herausforderungen, die beim Kombinieren von Wissensquellen in Form von Ontologien zu bewältigen sind. In Abbildung 4.2 sind die Probleme dargestellt, die Klein identifiziert und in drei Gruppen unterteilt. Das Kernproblem stellen die Unstimmigkeiten zwischen den Ontologien dar. Bei diesen wird zwischen Unstimmigkeiten auf Sprachebene und Unstimmigkeiten auf inhaltlicher Ebene unterschieden. Auf Sprachebene unterscheiden sich Ontologien in ihrer Syntax und ihrer Ausdrucksmächtigkeit. So können manche Ontologiesprachen Negation ausdrücken und andere nicht. Die Probleme auf **(1) Sprachebene** lassen sich laut Klein relativ einfach auflösen indem zum

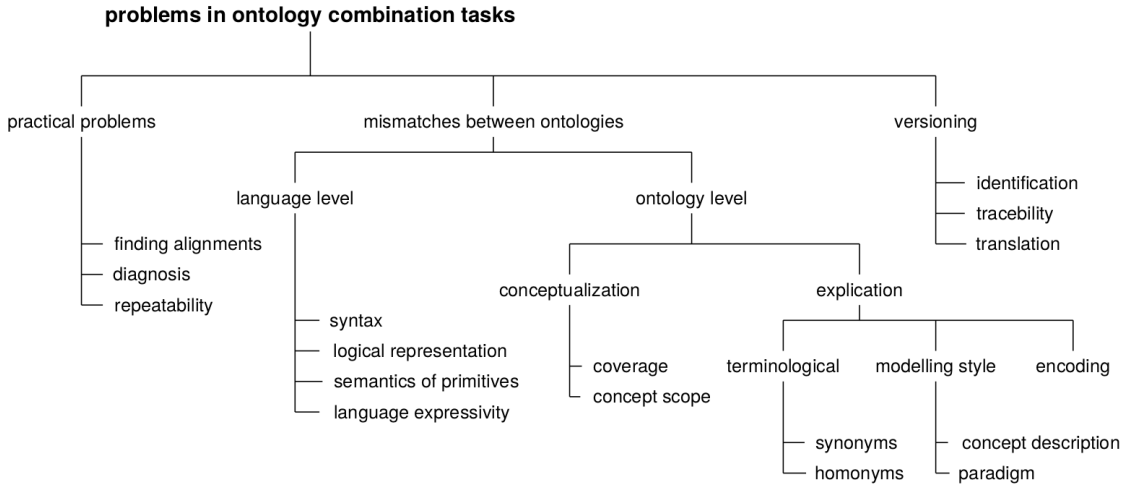


Abbildung 4.2: Übersicht über Probleme, die beim Kombinieren unabhängiger Ontologien auftreten können. Bildquelle: [Kle01]

Beispiel Regeln von einer logischen Darstellung in eine andere übersetzt werden. Dies verlangt eine genaue manuelle Betrachtung des Aufbaus einer logischen Darstellung, da die Elemente einer solchen Darstellung oft widersprüchlich benannt sind. Bei den Unstimmigkeiten auf inhaltlicher Ebene wird zwischen Unstimmigkeiten auf konzeptueller Ebene und Unstimmigkeiten auf expliziter Ebene unterschieden. Auf konzeptueller Ebene besteht das Problem des unterschiedlichen **(2.1) Umfangs der Konzepte**. Das bedeutet, dass zwei Konzepte aus zwei unterschiedlichen Ontologien vermeintlich das gleiche Konzept beschreiben, aber nicht genau die gleichen Instanzen haben. Ein Beispiel sind verschiedene Vorstellungen des Konzeptes „employee“ die zu unterschiedlicher Modellierung führen. Ein weiteres Problem auf konzeptueller Ebene ist die **(2.2) Modellabdeckung und Granularität**. So könnten in einer Ontologie nur Autos und keine Lastwägen enthalten sein, während eine Zweite beide modelliert und eine dritte unzählige weitere spezielle Untertypen von Autos und Lastwägen modelliert. Unterschiede in der Konzeptualisierung können laut Klein nicht automatisch aufgelöst werden, sondern benötigen Wissen und Entscheidungen eines Domänen-Experten. Die Modellabdeckung ist oftmals eher kein Problem sondern sogar der Anreiz überhaupt mehrere Quellen zu verwenden. Ein weiteres Problem ist die **(3.1) Art der Modellierung** von Konzepten. So könnte der Zusammenhang der Konzepte „dissertation“ und „publication“ zum Beispiel auf folgende Arten modelliert werden:

dissertation $<$ book $<$ scientific publication $<$ publication

dissertation $<$ scientific book $<$ book $<$ publication

dissertation $<$ scientific publication $<$ publication

dissertation $<$ book $<$ publication

Ein weiteres von Klein aufgeführtes Problem sind **(3.2) synonyme Terme**. So könnte eine Ontologie den Begriff „car“ nutzen, während eine andere Ontologie „automobile“ verwendet. Ein Thesaurus könnte hier bei der Auflösung von Synonymen helfen, wobei bei jeder Auflösung die Gefahr besteht den Umfang eines Konzeptes falsch zu wählen. Ein weiteres Problem auf dieser Ebene sind **(3.2) homonyme Terme**. Neben den genannten Unstimmigkeiten zwischen Ontologien nennt Klein die Versionierung von Ontologien als zentrale Herausforderung. Viele Wissensquellen wie Wikipedia entwickeln sich ständig weiter. Somit können über die Zeit Informationen auftreten die zuvor enthaltenem Wissen widersprechen. Es muss deshalb immer deutlich sein aus welcher Version einer Wissensquelle eine

Information stammt. Die dritte große Problemkategorie sind nach Klein die praktischen Probleme. Da Klein das vollständige Verbinden oder Angleichen von Ontologien für unrealistisch hält, ergibt sich ein hoher manueller Aufwand. Außerdem ist die Wiederholbarkeit von manuellen Schritten auf der nächsten Version der Wissensquellen nicht automatisch gewährleistet, sodass ein manueller Aufwand für jede neue Version einer Wissensquelle betrieben werden müsste. Klein stellt mit seiner Arbeit einen Rahmen zur Analyse der Machbarkeit für das verbinden von Wissensquellen. Er unterstreicht außerdem eine Unmöglichkeit des automatischen Verbindens.

Ein in Kleins Arbeit beschriebenes und in weiten Teilen der Literatur immer wieder erwähntes Werkzeug zum semi-automatischen Vereinigen von Ontologien ist PROMPT. In ihrer Arbeit **Algorithm and Tool for Automated Ontology Merging and Alignment** [NM00] beschreiben Noy und Musen dieses von ihnen entwickelte Werkzeug. PROMPT fungiert dabei als Erweiterung für das Ontologie-Werkzeug Protégé. Das semi-automatische Verfahren hinter PROMPT schlägt beim Vereinigen von Ontologien Kandidaten zum vereinigen von Konzepten vor. Die Vorschläge basieren dabei hauptsächlich auf syntaktischen Verbindungen zwischen Konzepten. Liegt also in einer Ontologie das Konzept „system“ vor und in einer anderen das Konzept „system“ mit dem gleichen Titel, so schlägt PROMPT eine Vereinigung dieser beiden Konzepte vor. Grundsätzlich ist das Werkzeug allerdings so aufgebaut, dass die Berechnung der Ähnlichkeit beliebig erweitert werden kann. Die Autoren schlagen beispielsweise vor das Informationen aus Thesauren hinzugezogen werden können, um Konzepte mit ungleichen aber synonymen Namen zu vereinigen. In ihrer Arbeit evaluieren Noy und Musen die durch PROMPT generierten Vorschläge beim Vereinigen zweier Ontologien mit insgesamt 134 Konzepten. Dabei wurden 74% der vom Nutzer ausgeführten Operationen durch PROMPT vorgeschlagen und 90% der von PROMPT vorgeschlagenen Operationen vom Nutzer ausgeführt. Eine detailliertere Betrachtung des Verfahrens hinter PROMPT wird in Abschnitt 5.5 im Rahmen der Untersuchung des Standes der Technik bei der Vereinigung von Ontologien vorgenommen.

4.2.2 Anreicherung von Hintergrundwissen aus unstrukturierten Wissensquellen

Neben den strukturierten Wissensquellen die Wissen meist in einem Graphen repräsentieren, gibt es auch unstrukturierte Wissensquellen. Diese enthalten Hintergrundwissen in Form von natürlichsprachlichen Sätzen. Die unstrukturierten Wissensquellen machen einen Großteil des vom Menschen niedergeschriebenen Wissens aus. Daher könnten sie für die lückenlose Erkennung und Einordnung, besonders von Domäne-Begriffen, notwendig sein.

In ihrer Arbeit **Enhancing Domain Knowledge for Requirements Elicitation with Web Mining** [KSY+10] liefern Kaiya et. al einen Ansatz um unstrukturiertes Wissen aus beliebigen Internetseiten in den Aufbau einer Wissensquelle einzubeziehen. Sie wollen dabei einen Beitrag zum Problem des Mangels an Wissensquellen mit Domänenwissen liefern. Dabei setzen sie auf eine Anreicherung bestehender Wissensquellen mit Konzepten und Beziehungen, die durch das Analysieren von Internetseiten gewonnen werden. Zunächst wird eine Domänen-Ontologie in mehrere Ontologien, die jeweils mehrere untergeordnete Domänen beschreiben, aufgeteilt. Dazu werden zentrale Konzepte über die Betweenness-Zentralität bestimmt. Daraufhin werden die zentralsten Konzepte jeweils mit ihren umliegenden Konzepten als neue Unter-Ontologie extrahiert. Für diese Unter-Ontologien wird im zweiten Schritt über klassische Suchmaschinen wie Google, nach passenden Dokumenten gesucht. Dazu werden die Konzepte der Unter-Ontologien zu einem Suchstring zusammengefügt. Im dritten Schritt werden Sätze in den gefundenen Dokumenten, die Konzepte aus der Unter-Ontologie enthalten, nach weiteren verbundenen Konzepten untersucht. Im letzten

Schritt werden die so ausgewählten Konzepte manuell, durch Experten, der Ontologie hinzugefügt. Zur Evaluation des Ansatzes wurde ein Experiment aufgesetzt, bei dem Personen mithilfe einer Ausgangs-Ontologie und einer mit dem vorgestellten Ansatz erweiterten Ontologie Anforderungen erheben sollten. Bei Verwendung der erweiterten Ontologien steigen die Vollständigkeit in zwei Experimenten von 24.4% auf 70.4% und von 31.9% auf 64.8% und die Richtigkeit von 69.2% auf 86.2% und von 87.9% auf 90.8% an.

5 Analyse und Entwurf

Diese Arbeit ist Teil des Projektes INDIRECT (siehe Kapitel 3) und soll im Rahmen des Projektes eine Grundlage schaffen um die Präzision der Anforderungsrückverfolgung (siehe Abschnitt 2.2.3), also der Verknüpfung von Anforderungen und Quelltextartefakten, zu verbessern. Ein großes Problem bei der Anforderungsrückverfolgung ist, dass eine syntaktische Verbindung zwischen den in den Anforderungen und den im Quelltext verwendeten Begriffen oftmals fehlt. Ein Grund hierfür kann sein, dass unterschiedliche Abstraktionsebenen für Konzepte in einer Anforderung und dem passenden Quelltextartefakt gewählt werden. Außerdem könnten verschiedene synonyme Begriffe für die gleichen Konzepte verwendet worden sein. Eine Möglichkeit Verknüpfungen dennoch korrekt herzustellen ist die Einbeziehung von Hintergrundwissen, um ein explizites Verständnis der Anforderungen zu erlangen. Solches Hintergrundwissen kann in Form von Konzepthierarchien (siehe Abschnitt 2.4.1) vorliegen, die den Zusammenhang verschiedener Konzepte (siehe Abschnitt 2.1.3) modellieren. Auf Basis dieser Modellierung kann dann eine Aussage über die Ähnlichkeit und den konkreten Zusammenhang von Konzepten getroffen werden. Außerdem kann sie Aufschluss über die Themenbereiche von Begriffen und damit von ganzen Dokumenten geben. Sind beispielsweise in einer Menge von Anforderungen besonders viele Begriffe in einer Konzepthierarchie mit Konzepten wie „interface“ oder „communication“ verbunden, so handelt es sich bei diesen Anforderungen möglicherweise um Beschreibungen von Softwareschnittstellen. Eine der bekanntesten Wissensquellen in der Computerlinguistik, die Informationen über semantische Beziehungen enthält, ist WordNet. Allerdings enthält, das von Lexikographen händisch erstellte, WordNet nur einen Teil der in Anforderungen vorkommenden Konzepte und damit die benötigten semantischen Informationen. Das liegt unter anderem daran, das die Basis für die in WordNet vorkommenden Worte Korpora wie der Brown Korpus sind, die vor allem Begriffe der Alltagssprache enthalten. Im Rahmen von INDIRECT werden jedoch vor allem Softwareanforderungen und Quelltextartefakte untersucht, die eine technischere Sprache als die Alltägliche enthalten können. Um eine fundierte Grundlage für die semantische Untersuchung von Konzepten in Anforderungen zu schaffen, müssten also vermutlich deutlich mehr als die in WordNet enthaltenen Informationen in die semantische Untersuchung von Anforderungen und Quelltext einbezogen werden. Ziel dieser Arbeit ist es daher einen Ansatz zu entwickeln mit dem flexibel weitere Wissensquellen eingebunden werden können und eine Konzepthierarchie aufgebaut werden kann, die die nötigen Informationen enthält um für möglichst alle in Softwareanforderungen vorkommenden Konzepte semantische Untersuchungen durchführen zu können.

In Abschnitt 5.1 wird dafür neben WordNet die aus Wikipedia abgeleitete Wissensquelle Wikidata vorgestellt. Da es in Wikipedia viele Artikel zu technischen Themen wie zum Beispiel bestimmten Rahmenwerken, oder häufig verwendeten Komponenten, wie einem „database management system“, aus der Softwareentwicklung gibt,kann es Sinn machen, eine solche Wissensquelle als Ergänzung zu WordNet hinzuzuziehen. Die weitere Analyse und der Entwurf, sowie die Evaluierung der Arbeit stützen sich hauptsächlich auf diese beiden Wissensquellen. Allerdings sollen sie flexibel austauschbar sein, sodass später leicht weitere Wissensquellen hinzugefügt werden können.

Damit mehrere Wissensquellen kombiniert werden können müssen, sie zunächst in ein einheitliches Format übersetzt werden. In Abschnitt 5.2 wird daher ein Modell für eine Konzepthierarchie entworfen, welches die benötigten semantischen Informationen enthält und in das sämtliche Wissensquellen übersetzt werden können.

Ein weiteres Teilproblem ist es, in beliebigen Wissensquellen das passende Konzept für einen Begriff in einem Kontext (siehe Abschnitt 2.1.4) zu finden. Da ein Begriff mehrere Bedeutungen haben kann (siehe Abschnitt 2.1.5), ist dies keine triviale Aufgabe. Abschnitt 5.3 beschäftigt sich deshalb mit der Bestimmung des passenden Konzeptes in einer beliebigen Wissensquelle auf Basis des Kontextes.

Auf Basis des angeglichenen Formats und der Möglichkeit für einen Begriff in einer Anforderung das passende Konzept in einer beliebigen Wissensquelle zu finden, soll dann eine Konzepthierarchie aufgebaut werden auf deren Basis die semantischen Beziehungen der Begriffe in Anforderungen untersucht werden können. Da ausgehend von einem gefundenen Einstiegskonzept unmittelbar auf die benachbarten Konzepte innerhalb der gleichen Wissensquelle zugegriffen werden kann, während für Schritte zu umliegenden Konzepten in einer anderen Wissensquelle zunächst eine Abbildung zu dieser notwendig ist, scheint es sinnvoll zu sein zunächst für jede Wissensquelle jeweils eine Konzepthierarchie aufzubauen. Dazu muss bestimmt werden welche Konzepte einer Wissensquelle für die spätere Weiterverarbeitung relevant sind und diese zu einer neuen Konzepthierarchie zusammengefügt werden. Dieser Aufbau einer Konzepthierarchie wird in Abschnitt 5.4 behandelt.

Nach der Erstellung einer Konzepthierarchie für jede Wissensquelle müssen die entsprechenden Informationen zusammengeführt werden. Enthält beispielsweise eine Wissensquelle ein Konzept A und nicht das Konzept B, während eine andere das Konzept B aber nicht das Konzept A enthält, kann für die Konzepte A und B bei getrennt aufgebauten Konzepthierarchien keine semantische Beziehung bestimmt werden. Abschnitt 5.5 beschäftigt sich deshalb mit der Frage wie die voneinander getrennten Informationen in den Konzepthierarchien kombiniert werden können.

5.1 Analyse existierender Wissensquellen

Kerngedanke dieser Arbeit ist es zwei syntaktisch unverbundene Begriffe mithilfe von Hintergrundwissen zu verbinden. Um zwei Begriffe die syntaktisch nicht gleich sind in Beziehung zu setzen benötigt man Wissen über deren Bedeutungen und deren Zusammenhang. Solche Zusammenhänge werden als semantische Relationen (siehe Abschnitt 2.1.6) bezeichnet. Kennt man die semantische Relation zweier ungleicher Begriffe, so weiß man zum Beispiel ob sie Synonyme sind oder ob einer der Begriffe ein Unterkonzept (Hyponym) des anderen darstellt. Mit diesem Wissen lassen sich also Aussagen darüber treffen ob zwei ungleiche Begriffe im Grunde das Gleiche, etwas Ähnliches, oder völlig verschiedene Dinge beschreiben. Diese Information kann einen wichtigen Beitrag zur Verknüpfung von Anforderungen und Quelltext leisten.

Für diese Arbeit sind also solche Wissensquellen relevant, die diese Arten von semantischen Beziehungen enthalten. Eine bekannte Art Wissen über Konzeptbeziehungen, wie sie bei

semantischen Relationen vorliegen, darzustellen sind sogenannte semantische Netzwerke. In einem semantischen Netzwerk sind Konzepte als Knoten modelliert und die Relationen zwischen den Konzepten als Kanten. Das gesamte semantische Netzwerk stellt also einen Graphen dar. Ein in der Computerlinguistik häufig verwendete Bezeichnung für ein solches semantisches Netzwerk ist der Begriff der Ontologie (siehe Abschnitt 2.4.2). Auch Text in natürlicher Sprache kann eine Wissensquelle für semantische Beziehungen sein. Aus diesem müssen die Zusammenhänge jedoch zunächst durch Textanalyse extrahiert werden.

5.1.1 WordNet

WordNet ist eine der bekanntesten lexikalischen Wissensquellen im Bereich der Computerlinguistik. Es wird seit 1985 kontinuierlich an der Universität Princeton weiterentwickelt. Die aktuellste Version 3.1 stammt aus dem Juli 2011. Anfangs wurden Begriffe der Alltagssprache zum Beispiel aus dem Brown Korpus als Grundlage für die Worte in WordNet gewählt, wobei später weitere Wortlisten als Grundlage hinzugekommen sind [Fel99]. Es enthält inzwischen 117.000 Konzepte und entsprechende semantische Beziehungen zwischen diesen. In den folgenden beiden Abschnitten wird betrachtet wie Konzepte und Beziehungen in WordNet aufgebaut sind.

5.1.1.1 Konzepte

Konzepte werden in WordNet als Synonymring (engl. Synset) dargestellt. Ein solcher Synonymring enthält eine Menge von Wortbedeutungen, die das gleiche Konzept beschreiben. Ein Wort kann mehrere Bedeutungen haben und somit in mehreren Synonymringen enthalten sein. Für jeden Synonymring gibt es eine Erläuterung des beschriebenen Konzeptes in Form eines kurzen Satzes. Diese wird in WordNet als Glosse (engl. gloss) bezeichnet. Zusätzlich können Synonymringe weitere Beispielsätze enthalten, die die Verwendung der enthaltenen Worte zeigen. Letztlich hat jeder Synonymring eine eindeutige Identifikationsnummer, welche sich aus der Zeilennummer im entsprechend hinterlegten Dokument aller Synonymringe einer Wortart ergibt (engl. offset). Beispiel 5.1 zeigt drei Konzepte, die eine Wortbedeutung des Wortes „user“ enthalten und die beschriebenen Elemente eines Konzeptes.

Beispiel 5.1: Konzepte, die eine Wortbedeutung von „user“ enthalten.

10761247 [user]:
„a person who makes use of a thing; someone who uses or employs something“

10092334 [exploiter, user]:
„a person who uses something or someone selfishly or unethically“

10055991 [drug user, substance abuser, user]:
„a person who takes drugs“

5.1.1.2 Relationen

Neben den Konzepten enthält WordNet Relationen. Dabei unterscheidet WordNet zwischen zwei Arten von Relationen. Zum einen enthält es semantische Beziehungen, die Konzepte miteinander verbinden und zum anderen lexikalische Beziehungen, die individuelle Worte miteinander verbinden. Im Rahmen dieser Arbeit sind vor allem die semantischen Beziehungen relevant, denn aus ihnen sollen später die Zusammenhänge zwischen Konzepten abgeleitet werden. In Tabelle 5.1 sind die verschiedenen Relationen zwischen Nomen

Tabelle 5.1: Übersicht über die Relationen zwischen Nomen in WordNet.

Relation	IDs	Definition	Beispiel
Hypernym	@	Von einem Konzept zum Übergeordneten	breakfast →meal
Hyponym	~	Von einem Konzept zum Untergeordneten	meal →lunch
Instance Hypernym	@i	Von einer Instanz zu ihrem Konzept	Austen →author
Instance Hyponym	i	Von einem Konzept zu einer Instanz	composer →Bach
Member Meronym	%m	Von einer Gruppe zu einem Mitglied	faculty →professor
Member Holonym	#m	Von einem Mitglied zu seiner Gruppe	copilot →crew
Part Meronym	%p	Von einem Ganzen zu einem Teil	table →leg
Part Holonym	#p	Von einem Teil zu einem Ganzen	course →meal
Substance Meronym	%s	Von Substanzen zu deren Teilen	water →oxygen
Substance Holonym	#s	Von Teilen einer Substanz zu dieser	gin →martini
Antonym	!	Semantisches Gegenteil	leader and follower
Derivationally Related Form	+	Lemmas mit gleicher morphologicher Wurzel	destruction and destroy

in WordNet aufgelistet [JM09]. Im Grunde umfassen sie die semantischen Relationen die in Abschnitt 2.1.6 vorgestellt wurden. Zu den Hyperonymen und Hyponymen kommen jeweils noch eine speziellere Beziehung hinzu, die die Relation zwischen einem Konzept und dessen Instanzen beschreibt. So ist die Beziehung *composer* → *Bach* ein Instanz Hyponym und die Beziehung *Austen* → *author* ein Instanz Hyperonym. Meronymie und Holonymie umfasst WordNet in jeweils drei Ausprägungen. Dabei wird jeweils zwischen Gruppen, Teilen und Substanzen unterschieden. So ist *water* → *oxygen* ein Substanz-Meronym, *table* → *leg* ein Teil-Meronym und $faculty \rightarrow professor$ ein Mitglied-Meronym.

5.1.2 Wikidata

Wikidata ist eine frei verfügbare Wissensquelle der Wikimedia-Stiftung. Sie wurde entwickelt um die unstrukturierten Informationen die sich in den Artikeln des bekannteren Schwesterprojektes Wikipedia verbergen zu strukturieren [VK14]. Dafür wurde zunächst aus jedem Wikipedia-Artikel ein Wikidata-Element erzeugt. Diese können wie Wikipedia-Artikel von jeder Person bearbeitet werden und miteinander in Beziehung gesetzt werden. Aus diesem Grund ist Wikidata sehr umfangreich aber auch fehleranfällig, da jederzeit Informationen ungeprüft verändert werden können. Zum Zeitpunkt der Arbeit enthält Wikidata laut offizieller Webseite knapp unter siebzig Millionen Elemente.

5.1.2.1 Konzepte

Die in Wikidata vorhandenen Elemente (engl. items) entsprechen der beschriebenen Vorstellung eines Konzeptes. Wie auch die Synonymringe in WordNet haben die Elemente in Wikidata eine eindeutige Identifikationsnummer (engl. item identifier) und einen Satz zur Beschreibung des Konzeptes (engl. description). Anders als in WordNet werden die einem Konzept zugrundeliegenden Worte und ihre Bedeutungen in Wikidata nicht explizit modelliert. Ein Wikidata-Element hat eine Bezeichnung (engl. label) das dem des zugrundeliegenden Wikipedia-Artikels entspricht. Neben dieser Hauptbezeichnung hat ein Wikidata-Element eine Menge von alternativen Bezeichnungen (engl. aliases). In Abbildung 5.1 ist beispielhaft das Konzept „Douglas Adams“ mit der Identifikationsnummer „Q42“ und seiner Beschreibung sowie seinen alternativen Bezeichnungen dargestellt. Beispiel 5.2 zeigt fünf verschiedene Wikidata-Elemente die den Begriff „user“ als Bezeichnung oder alternative Bezeichnung haben.

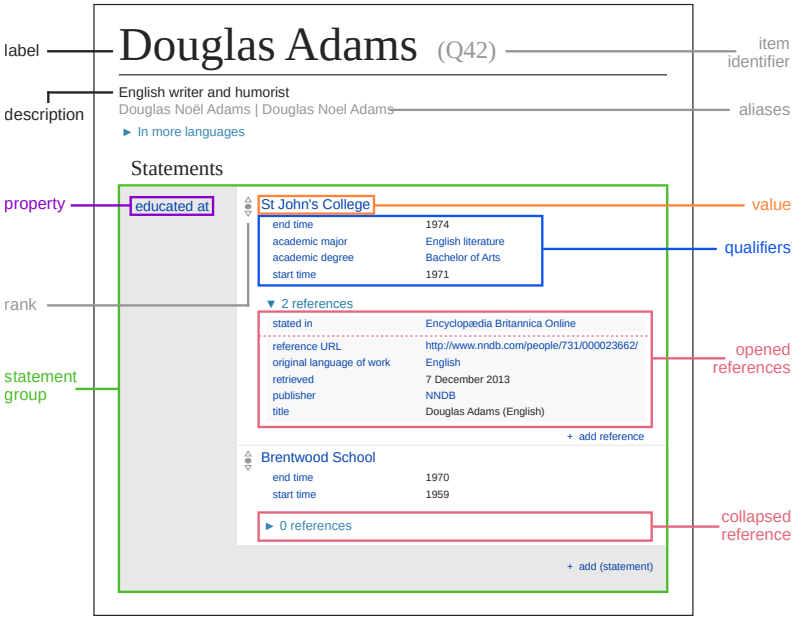


Abbildung 5.1: Übersicht über das Datenmodell von Wikidata.

Beispiel 5.2: Konzepte, die eine Wortbedeutung von „user“ enthalten.

Q278368 [user; computer user]:
„person who uses a computer or network service“

Q3604202 [user account; account, user]:
„entity representing a user of a computer system or network service“

Q26403005 [username; user account identifier, user name, user, user account name, screen name]:
„identifier for a person in a computer system“

Q613354 [user; system user]:
„person who interacts with a system, typically through an interface, to extract some functional benefit“

Q20850624 [User]:
„Ancient Egyptian nomarch“

5.1.2.2 Relationen

Relationen sind in Wikidata als Eigenschaft-Wert Paar abgebildet. Sie werden als Aussage (engl. statement) bezeichnet. Eine solche Aussage besteht aus einer Eigenschaft (engl. property) und einem Wert (engl. value). Eigenschaften sind eine eigene Art von Wikidata-Elementen. Werte können andere Wikidata-Elemente oder einfache Datentypen wie Anzahl, Zeitpunkt und geographische Koordinaten sein. Aufgrund dieser Flexibilität und Wikidatas Anspruch Wissen ganzheitlich abzubilden gibt es eine große Anzahl an Relationen in Wikidata. Die für diese Arbeit relevanten semantischen Beziehungen von Konzepten sind

Tabelle 5.2: Übersicht über die Relationen zwischen Elementen in Wikidata.

Relation	IDs	Definition	Beispiel
subclass-of	P279	Von einem Konzept zum Übergeordneten	breakfast →meal
instance-of	P31	Von einer Instanz zu ihrem Konzept	Austen →author
has-part	P527	Von einem Ganzen zu einem Teil	table →leg
part-of	P361	Von einem Teil zu einem Ganzen	course →meal
antonym	P5974	Semantisches Gegenteil	leader and follower

ebenfalls ein Teil dieser über sechstausend verschiedenen Eigenschaften, die ein Wikidata-Element haben kann. In Tabelle 5.2 sind die Wikidata-Eigenschaften aufgelistet, die den für diese Arbeit relevanten semantischen Beziehungen entsprechen. Bei der Betrachtung der Eigenschaften fällt auf, dass Beziehungen für Hyponymie und Instanz Hyponymie fehlen. Diese lassen sich lediglich aus den entsprechenden Hyperonymie-Beziehungen ableiten. Aus der Tatsache das „database“ das Überkonzept „collection“ hat kann abgeleitet werden, dass „collection“ das Unterkonzept „database“ besitzt. Besonderheit bei Wikidata ist außerdem, dass die Relationen von einzelnen Menschen oder von automatisierten Softwareanwendungen (engl. bots) gefolgert werden, sodass sie häufig unvollständig sind. Außerdem können fehlerhafte Beziehungen in Wikidata vorhanden sein. So ist zum aktuellen Zeitpunkt das Konzept des Kirchenfensters (Q64620757) als Unterkonzept des Konzeptes eines Fensters einer graphischen Benutzeroberfläche (Q835016) modelliert.

5.1.3 Weitere Wissensquellen

Im Rahmen dieser Arbeit wird vor allem mit den beiden vorgestellten Wissensquellen WordNet und Wikidata gearbeitet. Allerdings sollen alle Verarbeitungsschritte so entworfen werden, dass beliebige weitere Wissensquellen eingebunden werden können. Weitere Wissensquellen können sowohl allgemeine Wissensquellen wie CYC sein, als auch spezielle Domänen-Ontologien. Außerdem könnten aus unstrukturierten Informationen kleine semantische Netzwerke abgeleitet werden, die dann wiederum als Wissensquelle fungieren können.

5.2 Entwurf eines allgemeingültigen Datenmodells für eine Konzepthierarchie

Wie in Abschnitt 5.1 festgestellt, unterscheidet sich die Art auf die Konzepte und Relationen in verschiedenen Wissensquellen modelliert werden. Um mit beliebigen Wissensquellen einheitlich arbeiten zu können muss daher zunächst ein Modell zur Repräsentation einer Konzepthierarchie definiert werden. Ziel ist es dabei die Darstellungsarten der einzelnen Wissensquellen in diese einheitliche Darstellung zu übersetzten. Bezeichnet man die Wissensquellen als Ontologien wird dafür häufig der Begriff der Ontologie-Übersetzung (engl. ontology translation) verendet.

Aus den Gemeinsamkeiten der in Abschnitt Abschnitt 5.1 betrachteten Wissensquellen ergeben sich einige Attribute für die Modelle eines Konzeptes und einer Relation. Zunächst stellen beide Wissensquellen Konzepte dar. In WordNet geschieht das über so genannte Synonymringe, als einer Menge von Worten, die die gleiche Bedeutung teilen. In Wikidata liegen Konzepte als sogenannte Elemente vor. In beiden Wissensquellen liegen einem Konzept eine Menge von Worten zugrunde. In Wikidata können wir dabei die Bezeichnung eines Elementes und dessen alternative Bezeichnungen betrachten und in WordNet sind die dem Konzept zugrundeliegenden Worte explizit als solche modelliert. Der Grund

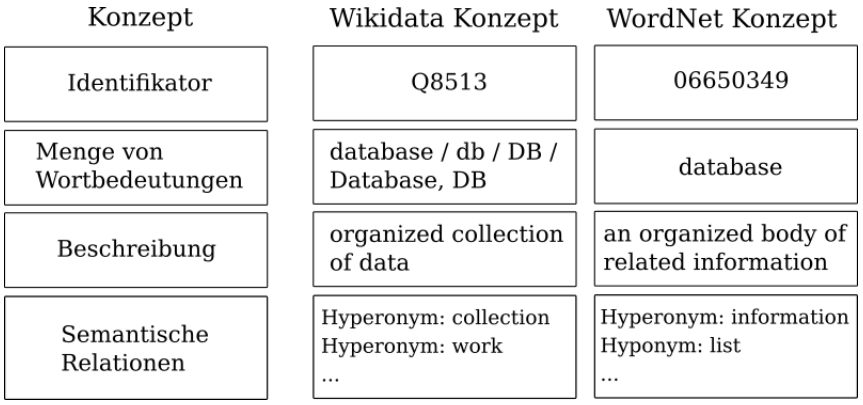


Abbildung 5.2: Modell eines Konzeptes und Abbildung eines Wikidata-Elementes und eines WordNet-Synsets auf dieses.

Tabelle 5.3: Abbildung der Relationen der Wissensquellen auf die Relationen im allgemeinen Modell. Die mit x^{-1} gekennzeichneten Relationen sind die in der jeweiligen Wissensquelle nicht vollständig vorhandenen inversen Relationen.

Allgemeines Modell	WordNet	Wikidata
Hyperonym	Hyperonym (@)	subclass-of (P279)
Hyponym	Hyponym (~)	$P279^{-1}$
Instanz Hyperonym	Instanz Hypernym (@i)	instance-of (P31)
Instanz Hyponym	Instanz Hyponym(~i)	$P31^{-1}$
Meronym	Meronym (%m, %p, %s)	has-part (P527), $P361^{-1}$
Holonym	Holonym (#m, #p, #s)	part-of (P361), $P527^{-1}$
Antonym	Antonym (!)	opposite-of (P461), $P461^{-1}$, antonym (P5974), $P5974^{-1}$

dafür, dass in Wikidata eines der Worte als Bezeichnung hervorgehoben wird, ist, dass die Wikidata-Elemente aus Wikipedia-Artikeln abgeleitet wurden und dabei zunächst nur deren Titel übernommen wurde. Erst später wurden die Konzepte um alternative Bezeichnungen ergänzt. Außerdem ist in beiden Wissensquellen eine Beschreibung für jedes Konzept vorhanden. In WordNet ist dies die Glosse (engl. gloss) und in Wikidata die Beschreibung (engl. description). In Abbildung 5.2 sind die Elemente eines allgemeinen Konzeptes und die konkreten Ausprägungen für Wikidata und WordNet beispielhaft dargestellt.

Neben diesen notwendigen Attributen für ein Konzept, sind weitere ergänzende Attribute für das Datenmodell denkbar. So existieren zum Beispiel für die Konzepte in Wikidata oft Referenzen auf deren Wikipedia Artikel. Solche zusätzlichen Informationen zu Konzepten können neben der Beschreibung den Kontext (siehe Abschnitt 2.1.4) eines Konzeptes ergänzen. Der Kontext ist ein wichtiger Teil für die Disambiguierung (siehe Abschnitt 2.3.1), also der Auflösung von Mehrdeutigkeiten, die in dieser Arbeit immer wieder vollzogen werden muss. Es kann also Sinn machen den Kontext als eigenes Attribut in Form einer Menge an Worten die mit dem Konzept in Verbindung stehen, zu modellieren.

Auch Relationen zwischen den Konzepten werden in beiden Wissensquellen modelliert. Diese bestehen in beiden Fällen aus einem Typ und einer Richtung. Um die Relationen in ein einheitliches Modell zu überführen, muss zunächst ein einheitlicher Katalog an Relationen aufgestellt werden und danach die Relationen der Wissensquellen auf die Relationen im Katalog abgebildet werden. Als die für diese Arbeit relevanten Relationen wurden be-

reits die semantischen Relationen (siehe Abschnitt 2.1.6) identifiziert. In der Tabelle 5.3 sieht man die grundlegenden semantischen Relationen und die Identifikatoren ihrer entsprechenden Vertreter in WordNet und Wikidata. Für die Zuordnung wurde in WordNet entsprechend der gleichen Bezeichner der Relationen abgebildet und die verschiedenen Typen von Meronoymie und Holonymie auf den Grundtyp abgebildet. Für Wikidata wurden mit dem Werkzeug zum finden der verschiedenen Beziehungstypen nach den Bezeichnern gesucht und mögliche passende Relationen identifiziert. Dabei muss darauf geachtet werden, dass in Wikidata häufig die inverse Relation fehlt und daher vor der Übertragung ins allgemeine Datenmodell ergänzt werden muss. Die inverse Relation ist dabei für eine Hyperonymie-Beziehung eine Hyponymie-Beziehung und für eine Meronymie-Beziehung eine Holonymie-Beziehung. Synonymie als Inverses der Antonymie muss nicht modelliert werden, da synonyme Konzepte einfach zu einem zusammengefügt werden können.

5.3 Bestimmung der Einstiegskonzepte

Nachdem im vorangegangenen Abschnitt ein Datenmodell entwickelt wurde, in das Konzepte und Relationen der Wissensquellen übersetzt werden können, stellt sich nun die Frage wie die relevanten Einstiegskonzepte zur Konsolidierung gefunden werden können. Mit einem Einstiegskonzept ist das Konzept in einer Wissensquelle gemeint, das dem zu untersuchenden Konzept entspricht. Befindet sich also in einer Anforderungsbeschreibung im Kontext von Software die Entität „Dashboard“ so ist das Konzept „dashboard: control panel of a software application“ für Wikidata das passende Einstiegskonzept. Ausgangspunkt für die Entitäten aus Anforderungsdokument und Quelltext für die Einstiegskonzepte gefunden werden müssen sind in INDIRECT die vom Entitätenerkenner (EntityRecognizer) erkannten Entitäten. Diese werden als Knoten in den IGraph geschrieben. Der Entitäten-Knoten (contextEntity) hat als Attribut den Namen der Entität also zum Beispiel „dashboard“. Außerdem ist der Entitäten-Knoten mit den Token-Knoten verbunden, auf denen die Entität basiert. Im Falle der Entität „dashboard“ wäre das nur ein Token mit dem Namen „dashboard“. Die Token enthalten neben dem Namen auch Informationen über das vorangegangene und das folgende Wort. Somit können über eine Betrachtung der umliegenden Token-Knoten die Worte gesammelt werden, die im Satz um die Entität herum vorkommen. Sie ergeben also den Kontext der Entität. Dies ist notwendig um die Einstiegskonzepte für homonyme Begriffe zu finden. Für das Beispiel „dashboard“ kennt Wikidata mehrere verschiedene Konzepte. Unter anderem sind das die Konzepte „dashboard: control panel of a software application“ und „/dashboard: control panel located directly ahead of a vehicle's driver, displaying instrumentation and controls for the vehicle's operation“. Zusätzlich kennt Wikidata ein Lied mit dem Namen „dashboard“. WordNet kennt dagegen für den Begriff „dashboard“ nur das Konzept im Sinne eines Teils im Auto und nicht das der Oberfläche einer Softwareanwendung. Um die auftretenden Entitäten also sinnvoll mit Hintergrundwissen anzureichern müssen zunächst die passenden Konzepte in den Wissensquellen gefunden werden.

Diesen Vorgang der Abbildung von einem Wort und dessen Kontextes auf ein Konzept nennt man in der Computerlinguistik Wort-Sinn-Disambiguierung (engl. Word Sense Disambiguation oder WSD) (siehe Kapitel 2). Einen Überblick über mögliche Ansätze zur Wort-Sinn-Disambiguierung geben Daniel Jurafsky und James H. Martin in ihrem Buch **Speech and Language Processing** [JM09]. Als erste Möglichkeit für eine WSD stellen sie Ansätze vor, die auf überwachtem Lernen (engl. supervised learning) basieren. Das Problem dieser Ansätze ist, das für jede Wissensquelle ein neuer Klassifikator trainiert werden muss. Dafür müssen zunächst entsprechende Trainingsdaten manuell erstellt werden. Während generell die Entwicklung solcher Modelle für bestimmte populäre Wissensquellen eine valide Möglichkeit ist um die Einstiegskonzepte im Rahmen von INDIRECT zu finden, muss

es jedoch immer auch eine Möglichkeit geben eine Wissensquelle zu verwenden, falls entsprechend trainierte Modelle nicht vorhanden sind. Dies ist gerade vor dem Hintergrund wichtig, dass es Idee dieser Arbeit ist, jede beliebige Wissensquelle mit möglichst geringem Aufwand zu INDIRECT hinzufügen zu können. Passender für die Bestimmung der Einstiegskonzepte im Rahmen von INDIRECT scheinen daher die von Jurafsky und Martin vorgestellten Wörterbuch und Thesaurus basierten Methoden. Bekanntester Vertreter ist dabei der Lesk-Algorithmus. Auf ihm basiert eine ganze Familie von Algorithmen, die zur Bestimmung der richtigen Wortbedeutung für einen Begriff die maximale Überlappung der Worte in der Umgebung des Begriffes und der Worte in der Beschreibung des Konzeptes in der Wissensquelle verwenden. Auch die beiden Ansätze zur Abbildung von Konzepten zwischen verschiedenen Wissensquellen im Rahmen der Arbeit zur Konstruktion von BabelNet (siehe Kapitel 4) basieren im Wesentlichen auf dem im Folgenden dargestellten Lesk-Algorithmus:

Algorithmus 1 Grundstruktur des Lesk-Algorithmus

```
bestSense ← most frequent sense of word
maxOverlap ← 0
context ← set of words in sentence
for each sense ∈ senses of word do
    signature ← set of words in the gloss and examples of sense
    overlap ← ComputeOverlap(signature, context)
    if overlap > maxOverlap then
        maxOverlap ← overlap
        bestSense ← sense
    end if
end for
```

Darauf aufbauend sind verschiedene Optimierungsmöglichkeiten denkbar. So werden in [NP12] neben den Glossen der möglichen Zielkonzepte auch die Namen und Glossen der Umgebungskonzepte in die Berechnung der Überlappung mit einbezogen. Bei diesem Vorgang muss je nach Umfang der ausgewählten Umgebungskonzepte zwischen dem Übersehen von Überlappungen und dem Einbeziehen irrelevanter Überlappungen abgewogen werden. Als weitere Optimierungsmöglichkeit nennen Jurafsky und Martin die Gewichtung der Begriffe in den Kontexten. Dazu könnte zum Beispiel wieder das Tf-idf-Maß (siehe Kapitel 2) verwende werden, wobei dafür die Auftrittshäufigkeit für Begriffe in Beschreibungen einer Wissensquelle bekannt sein müsste.

Neben den Optimierungsmöglichkeiten muss, beim Entwurf eines auf dem Lesk-Algorithmus basierenden Verfahrens außerdem ein grundsätzliches Problem beachtet werden. Der Lesk-Algorithmus ist in seiner Grundform darauf ausgelegt das passendste Konzept zu finden. Da aber die in Anforderungsdokumenten auftretenden Entitäten oftmals sehr technisch sind können verstärkt Situationen auftreten in denen ein Teil der Wissensquellen eine technische Bedeutung eines Begriffes nicht kennen. So wurde in einem vorangegangenen Beispiel bereits darauf hingewiesen, dass WordNet für den Begriff „dashboard“ nur die Bedeutung im Sinne eines Teiles im Auto und nicht die aus dem Bereich der Software kennt. Eine einfache Maximierung der Schnittmenge zweier Kontexte reicht nicht aus um zu erkennen, dass das am besten passende Konzept dennoch nicht das richtige ist. Eine mögliche Lösung ohne eine grundlegende Veränderung des Verfahrens ist hier die Einführung von einem Mindestwert an Überschneidung der Kontexte. Dieser Ansatz könnte auf der Annahme beruhend funktionieren, dass die Kontexte von Begriff und Konzept einer Wissensquelle allgemein eine konstant hohe durchschnittliche Anzahl an Überschneidungen haben. Variiert die Überschneidung der Kontexte allerdings stark und ist nicht weit genug von einem möglichen Grenzwert entfernt, besteht die Gefahr viele berechtigte Di-

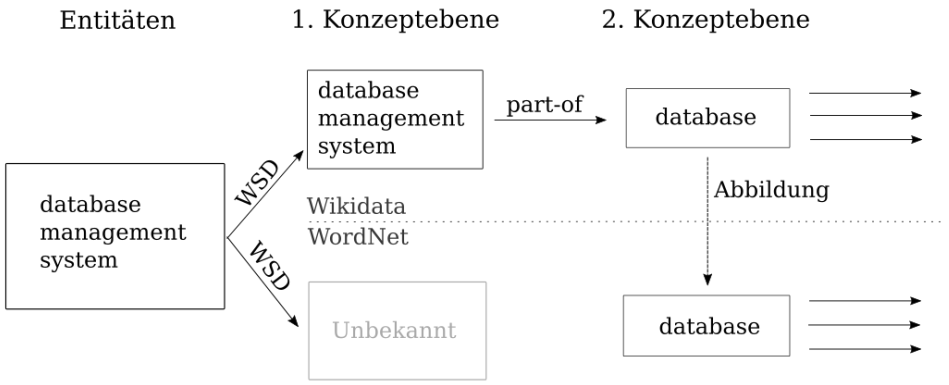


Abbildung 5.3: Finden von Einstiegskonzepten in der zweiten Konzeptebene durch die Kombination von Wissensquellen.

sambiguierungsergebnisse aufgrund des Schwellwertes zu verwerfen. Wie gut ein solcher Schwellwert das Ergebnis der Bestimmung der Einstiegskonzepte verbessern kann soll in der Evaluation gezeigt werden.

Nach der Bestimmung der Einstiegskonzepte aller Wissensquellen für jede Entität, können pro Entität drei Fälle auftreten:
Fall eins beschreibt den Fall, dass für jede Wissensquelle ein passendes Einstiegskonzept gefunden wurde. Dieser stellt den optimalen Fall dar, bei dem keine weitere Verarbeitung nötig ist. Allerdings basiert diese Arbeit auf der, dass sich die ausgewählten Wissensquellen inhaltlich ergänzen. Gerade wenn kleine Domänen-Ontologien einbezogen werden, können also vermehrt Fälle auftreten bei denen zumindest für diese ein großer Teil der gesuchten Konzepte nicht enthalten ist. Deshalb ist es wichtig auch die anderen Fälle entsprechend zu behandeln.

Fall zwei beschreibt den Fall das mindestens eine aber nicht alle Wissensquellen ein passendes Konzept gefunden haben. Dies ist bei Verwendung mehrerer sich ergänzender Wissensquellen häufig zu erwarten. In diesem Fall kann der beschriebene Ansatz zum Finden eines Einstiegskonzeptes statt direkt auf die Entität, auf einem Hyperonym des gefunden Einstiegskonzeptes einer anderen Konzepthierarchie ausgeführt werden. So kann für eine Konzepthierarchie, die einen detaillierten Begriff nicht enthält ein Einstiegskonzept auf Basis eines allgemeineren Begriffes gefunden werden. In Abbildung 5.3 ist ein Beispiel für diesen Fall abgebildet. In diesem Fall ist in der Wissensquelle WordNet kein Konzept für die Entität „database management system“ vorhanden, in Wikidata dagegen schon. In Wikidata ist das Konzept des „database manegement systems“ mit dem Konzept „database“ über eine semantische Relation verbunden. Das Konzept „database“ ist wiederum auch in WordNet vorhanden. Durch eine Abbildung des Konzeptes aus Wikidata auf das passende Konzept in WordNet, kann also eine indirekte Verbindung der Entität „database management system“ in den WordNet Konzeptgraphen gefunden werden. Durch diese Verbindung wiederum können zusammenhänge abgeleitet werden, die sonst nicht erfasst worden wären.

Der **dritte Fall** der auftreten kann ist der, das keine der Wissensquellen ein passendes Konzept für eine Entität kennt. Dies scheint vor allem dann der Fall zu sein wenn Entitäten entweder komplexe zusammengesetzte Begriffe sind oder unbekannte Abkürzungen und Wortneuschöpfungen. Um Wortneuschöpfungen und eigenen Abkürzungen der Autoren der Anforderungsdokumente richtig zuordnen zu können, müsste man zunächst das

Anforderungsdokument untersuchen. Die IEEE Recommended Practice for Software Requirements Specifications [EaEE98] schlagen vor zu Beginn eines Anforderungsdokumentes eigene Definitionen, Abkürzungen und Akronyme zu erklären. Es macht also perspektivisch Sinn derartige Zusatzinformationen in INDIRECT einzubeziehen um entsprechende Entitäten zu erkennen.
Konnte für eine Entität jedoch in keiner Wissensquelle ein Konzept gefunden werden, da die Entität ein zusammengesetzter Begriff ist, könnte man diesen reduzieren um doch noch passende Konzepte zu finden. Die Idee ist, zusammengesetzte Begriff wie „security **system**“ oder „**setting** of system“ auf den sogenannten lexikalischen Kopf (siehe Abschnitt 2.1.8) zu reduzieren. Dieser beschreibt in den meisten Fällen die gleiche Entität in einer weniger spezifischen Form. Für allgemeine Wissensquellen ist die Wahrscheinlichkeit einen solchen allgemeineren Begriff zu kennen wesentlich höher als die einen komplexen, zusammengesetzten Begriff zu kennen. Li und Cleland-Huang verwenden in ihrer Arbeit (siehe **??**) folgende drei Regeln um den lexikalischen Kopf aus einer Nominalphrase zu extrahieren:

$$\begin{aligned} &1.\ Fall : [noun, verb]^* \ \textbf{<noun>} \\ &2.\ Fall : [noun, verb]^* \ \textbf{<noun>} < prep > < noun > \\ &3.\ Fall : [noun, verb]^* \ \textbf{<noun>} < attributive\ clause > \end{aligned} \tag{5.1}$$

Das fett gedruckte Nomen beschreibt jeweils den lexikalischen Kopf der Entität. Der erste Fall bei dem der lexikalische Kopf das letzte Nomen einer Nominalphrase ist stell den häufig auftretenden Normalfall dar. Der zweite Fall betrifft die Situation in der zwei Nomen durch eine Präposition getrennt sind. Hier ist das vor der Präposition stehende Nomen der lexikalische Kopf. Ein häufig auftretendes Beispiel ist die Präposition „of“. Im Beispiel „setting of a system“ ist der Teil der Phrase nach der Präposition eine Spezifizierung der eigentlichen Sachen, die durch das Nomen vor der Präposition beschrieben wird. Der dritte Fall beschreibt den Fall bei dem ein Nomen durch einen Relativsatz spezifiziert wird. Dieser Fall tritt allerdings bei den im Rahmen dieser Arbeit zu untersuchenden Entitäten nicht auf. Li und Cleland-Huang nutzen die beschriebenen Regeln um unmittelbar den lexikalischen Kopf zu bestimmen und diesen als Schlüsselterm in ihre Berechnung einbeziehen. Eine Zerteilung in mehrere Terme eines zusammengesetzten Ausdrucks findet in ihrer Arbeit auf Basis von Token statt. Da der Ansatz darauf basiert eine Term-zu-Phrasen Ähnlichkeit zu bestimmen, ist die Rolle eines Terms in einer Phrase in deren Ansatz nur bedingt entscheidend. Für die Entität „security protocol of system“ würden Li und Cleland-Huang also einfach die Term-zu-Phrasen Ähnlichkeit der drei Nomen in der Entität zu einer Zielphrase berechnen und die Term-zu-Phrasen Ähnlichkeit für „protocol“ höher gewichten, da es sich um den lexikalischen Kopf handelt. Im Rahmen dieser Arbeit sollen jedoch die tatsächlichen Entitäten die vom EntityRecognizer (siehe Abschnitt 3.2) bestimmt werden in einer Konzepthierarchie verbunden werden. Würde man die einzelnen Terme in der Entität als Einstiegspunkte in die Konzepthierarchie verwenden, könnte man nicht mehr die semantische Relation der ursprünglichen Entität ableiten. Das liegt daran, dass nur der lexikalische Kopf eine Entität als das was sie im einfachsten Sinne ist beschreibt. Die beschreibenden Nomen helfen zwar eine Entität zu spezifizieren, beschreiben aber nicht die gleiche Sache. Deshalb ist ihr Einstiegspfad in eine Konzepthierarchie von dem der eigentlichen Entität unabhängig. Aus dieser Beobachtung heraus ergibt sich eine Möglichkeit zusammengesetzte Begriffe mit Konzepten in Wissensquellen zu verbinden, indem man sie auf ihren lexikalischen Kopf reduziert. Aus der Definition des lexikalischen Kopfes (siehe Abschnitt 2.1.8) ergibt sich, dass ein zusammengesetzter Begriff ein Konzept beschreiben muss, das ein Unterkonzept des Konzeptes ist, das sein lexikalischer Kopf beschreibt.

Auf dieser Basis wird ein Verfahren zur Bestimmung von Einstiegskonzepten und abgeleiteten Konzepten auf Basis einer Nominalphrase vorgeschlagen. In Abbildung 5.4 wird

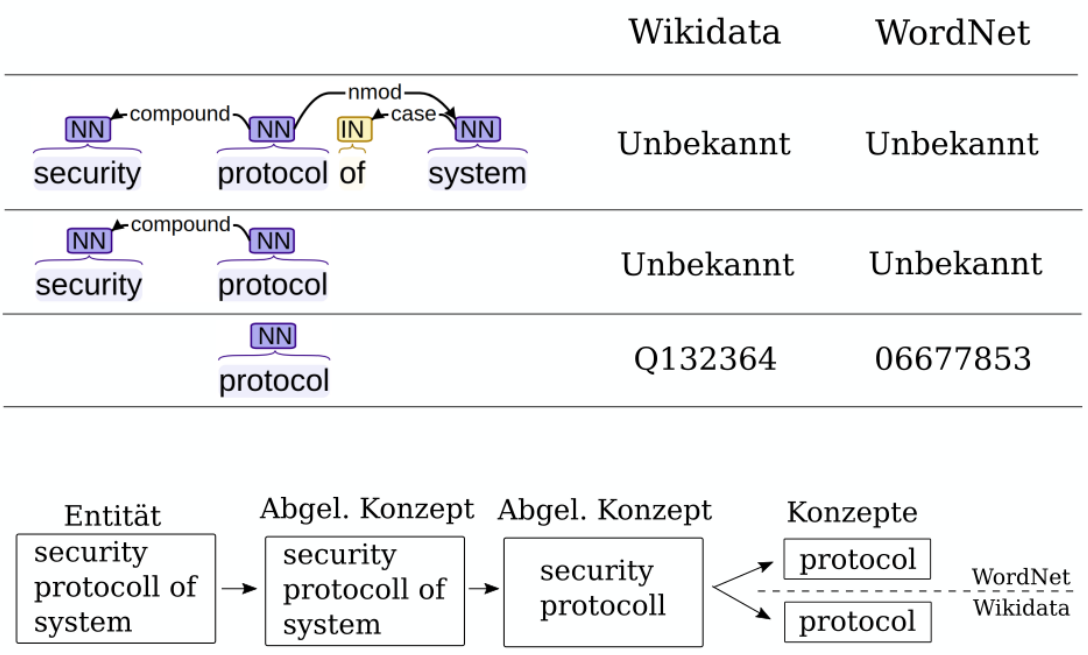


Abbildung 5.4: Durch die Reduzierung von Entitäten können Konzepte für eine allgemeinere Entität gefunden werden.

dessen Funktionsweise beispielhaft anhand der Phrase „security protocol of system“ dargestellt. Dabei wird zunächst geprüft ob die Nominalphrase eine Präposition enthält. Ist dies der Fall, werden die Worte vor der Präposition als reduziere Version der ursprünglichen Nominalphrase betrachtet. Auf diese wird vom ersten Wort an solange ein Nomen entfernt bis nur noch der das letzte übrig ist. Dieses stellt den lexikalischen Kopf dar. Die Schritte zur Reduzierung werden solange ausgeführt, bis eine der Wissensquellen ein Konzept für den aktuellen Reduktionsschritt enthält. Die reduzierten Formen einer Nominalphrase für die keine Wissensquelle ein Konzept kennt werden als abgeleitete Konzepte betrachtet. Zwischen ihnen wird eine Hyperonym-Relation zum abgeleiteten Konzept für eine allgemeinere Nominalphrase angenommen.

5.4 Aufbau von Konzepthierarchien

Wie in der Einleitung des Kapitels beschrieben, sollen zunächst unabhängige Konzepthierarchien für die jeweiligen Wissensquellen aufgebaut werden. Innerhalb einer Wissensquelle kann über die entsprechenden Relationen unmittelbar die Beziehung zwischen Konzepten bestimmt werden. Im Grunde könnte man von den gefundenen Einstiegskonzepten ausgehend die gesamte Wissensquelle als Konzepthierarchie für weitere Verarbeitungsschritte speichern. Allerdings ist das bei großen Wissensquellen unpraktikabel und nicht nötig da die benötigten Konzepte meist nur einen kleinen Teil aller Konzepte in einer Wissensquelle ausmachen. Es macht also Sinn nur einen Teil der Wissensquelle als Konzepthierarchie zu übernehmen. Welche Konzepte sinnvollerweise zu der Konzepthierarchie hinzugefügt werden hängt dabei vom Anwendungsfall dieser ab.

Im Rahmen dieser Arbeit wurden zwei mögliche Anwendungsfälle von Konzepthierarchien für INDIRECT identifiziert. Zum einen kann eine Konzepthierarchie Aufschluss darüber geben, wie zwei Konzepte zusammenhängen. Dieser Fall wird in Abschnitt 5.4.1 behandelt. Zum anderen kann eine Konzepthierarchie Aussagen darüber treffen, wie spezifisch ein

Konzept ist und zu welchem Themenkomplex es gehört. Dieser Anwendungsfall wird in Abschnitt 5.4.2 beschrieben.

5.4.1 Semantischer Zusammenhang von Konzepten

Eine grundlegende Funktion einer Konzepthierarchie ist es die semantischen Zusammenhänge zweier Konzepte zu betrachten. Als Einstiegsbeispiel dieser Arbeit wurde in Kapitel 1 dargestellt, wie durch eine semantische Verbindung zwischen Konzepten die in Anforderungen und Konzepten die im Quelltext vorkommen ein Beitrag zur Anforderungsrückverfolgbarkeit (siehe Abschnitt 2.2.3) geleistet werden kann. In [JM09] werden einige grundlegende Techniken zu Bestimmung des semantischen Zusammenhangs zweier Konzepte beschrieben:

Die einfachste Methode ist die Bestimmung der Anzahl der Kanten des kürzesten Pfades zwischen zwei Konzepten:

$$pathlen(c_1, c_2) = Anzahl\ der\ Kanten\ im\ \text{kürzesten}\ Pfad\ zwischen\ c_1\ und\ c_2 \tag{5.2}$$

Dieser Ansatz basiert auf der Annahme, dass die direkten Nachbarn eines Konzeptes in einem semantischen Netzwerk wesentlich enger mit ihm verwand sind als andere Konzepte die an entfernteren Stellen im semantischen Netzwerk vorliegen. Eine Variante der Länge des kürzesten Pfades ist dessen logarithmische Betrachtung:

$$sim_{path}(c_1, c_2) = -log(pathlen(c_1, c_2)) \tag{5.3}$$

Die Schwäche dieses Ansatzes ist, dass er auf der Annahme beruht, dass jede Kante im Netzwerk das gleiche Maß an Distanz ausdrückt. Tatsächlich ist dies aber oft nicht der Fall. Andere Möglichkeiten zur Berechnung des semantischen Zusammenhangs zweier Konzepte beruhen auf dem Informationsgehalt von Konzepten. Der Informationsgehalt $P(c)$ eines Konzepte berechnet sich als Wahrscheinlichkeit, dass ein zufällig ausgewähltes Wort ein Unterkonzept von diesem ist:

$$P(c) = \frac{\sum_{w \in words(c)} count(w)}{N} \tag{5.4}$$

Ein weiterer Bestandteil von fortgeschritteneren Ansätzen ist die Bestimmung des kleinsten gemeinsamen Überkonzeptes (engl. least common subsumer oder LCS) zweier Konzepte:

$$\begin{aligned} LCS(c_1, c_2) = {} & niedrigster\ Knoten\ der\ Hierarchie \\ & der\ \text{Überkonzept}\ von\ sowohl\ c_1\ als\ auch\ c_2\ ist \end{aligned} \tag{5.5}$$

Die Resnik-Ähnlichkeit berechnet sich darauf aufbauend als der logarithmisierte Informationsgehalt des LCS:

$$sim_{resnik}(c_1, c_2) = -log(P(LCS(c_1, c_2))) \tag{5.6}$$

Weitere Ähnlichkeitsmaße sind die Lin-Ähnlichkeit, die die Resnik-Ähnlichkeit mit den Informationsgehalten der Konzepte c_1 und c_2 gewichtet

$$sim_{Lin}(c_1, c_2) = \frac{2log(P(LCS(c_1, c_2)))}{log(P(c_1)) + log(P(c_2))} \tag{5.7}$$

und die Jiang-Conrath Distanz, die von der Resnik-Ähnlichkeit die Summe der Informationsgehalte von c_1 und c_2 abzieht:

$$dist_{JC}(c_1, c_2) = 2log(P(LCS(c_1, c_2))) - (log(P(c_1)) + log(P(c_2))) \tag{5.8}$$

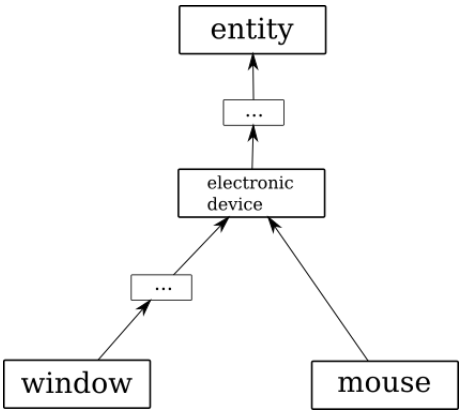


Abbildung 5.5: Durch die Pfad zwischen Konzepten und der Wurzel in der Konzepthierarchie ergeben sich Themen.

Ein weiteres Ähnlichkeitsmaß ist die erweiterte Lesk-Ähnlichkeit, die auf den Beschreibungen zweier Konzepte und der Konzepte in ihrer unmittelbaren Umgebung basiert. Sie wird als Schnittmenge der Worte in den Glossen von A und B, sowie den Glossen in ihrer unmittelbarer Umgebung hinsichtlich einer beliebigen Menge von Relationen bestimmt.

Grundsätzlich sollte es das Ziel der aufbauten Konzepthierarchie sein die zur Berechnung der Ähnlichkeiten zweier Konzepte nötigen Informationen zu enthalten. Ist das der Fall könnten auf Basis der erzeugten Konzepthierarchie schnell und einfach die Zusammenhänge der enthaltenen Konzepte berechnet werden. Die zur Berechnung der erwähnten Ähnlichkeitsmaße nötigen Informationen sind dabei der kürzeste Pfad zwischen zwei Konzepten, deren LCS, die Beschreibungen der Konzepte und ihrer unmittelbaren Nachbarn und die Informationsgehalte der Konzepte sowie des LCS. Daraus ergibt sich ein Bedarf an Knoten der aus den ursprünglichen Wissensquellen in die Konzepthierarchie hinzugefügt werden muss. Zum einen sollen alle Konzepte hinzugefügt werden, die direkt mit einem Einstiegskonzept in der Konzepthierarchie verbunden sind. Daraus lässt sich die erweiterte Lesk-Ähnlichkeit zwischen beliebigen Paaren von Einstiegskonzepten in der erstellten Konzepthierarchie berechnen. Des weiteren sollen alle Knoten der kürzesten Pfade zwischen allen Paaren von Einstiegskonzepten zur Konzepthierarchie hinzugefügt werden. Diese ergeben wie oben beschrieben ein eigenes Maß an Ähnlichkeit. Letztlich soll der LCS aller Paare von Einstiegskonzepten zur Konzepthierarchie hinzugefügt werden.

5.4.2 Themenextraktion für Konzepte

Neben den direkten semantischen Zusammenhängen zwischen zwei Konzepten kann in einer Konzepthierarchie vor allem auch der Zusammenhang mehrerer Konzepte in einem Themenkomplex untersucht werden. Dabei können alle Konzepte die Unterkonzept eines Konzeptes T sind als dem Thema T zugehörig betrachtet werden. Alle Konzepte die also Unterkonzept des Konzeptes $Software$ sind haben in irgendeinem Sinne mit dem Thema Software zu tun. Dieser Zusammenhang wurde bereits in verschiedenen Arbeiten dazu genutzt um für natürlichsprachliche Texte die Zugehörigkeit zu bestimmten Themen zu analysieren und dadurch eine Such- und Filtermöglichkeit für diese Texte zu bieten. In ihrer Arbeit **Automating creation of hierarchical faceted metadata structures** [SHR07] beschreiben Stoica et al. einen solchen Ansatz. Im Grund verbinden sie die in Texten auftretenden Konzepte mit der Wurzel der Konzepthierarchie, welche im Falle von WordNet das Konzept „entity“ ist und entnehmen den geteilten Pfaden zur Wurzel eine gemeinsame Themenzugehörigkeit der Konzepte. In Abbildung 5.5 ist beispielhaft dargestellt wie

sich für die Konzepte „window“ und „mouse“ durch deren Pfade zu „entity“ das gemeinsame Überkonzept „electronic device“ ergibt. Dieses Überkonzept stellt ein Thema dar, zu dem die darunter liegenden Konzepte gehören. Darauf aufbauend wird die dadurch entstandene Konzepthierarchie komprimiert, sodass aus den Knoten direkt die Themen abgelesen werden. Während die genaue Funktionsweise dieses und ähnlicher Ansätze im Rahmen dieser Arbeit nicht relevant ist, kann festgehalten werden, dass eine Verbindung von jedem Einstiegskonzept zum Wurzelknoten der jeweiligen Wissensquelle eine wichtige Information zur Bestimmung von Themen ist. Es soll daher für jedes Einstiegskonzept in der zu erstellenden Konzepthierarchie der Pfad entlang der Überkonzeptbeziehungen bis zum Wurzelknoten der Konzepthierarchie hinzugefügt werden.

5.4.3 Entwurf des Aufbaus einer Konzepthierarchie

In den beiden letzten Abschnitten wurde untersucht welche Informationen zu einer Konzepthierarchie hinzugefügt werden müssen, damit grundlegende Ähnlichkeitsmaße auf ihnen berechnet werden können und sich Themen extrahieren lassen. Daraus ergibt sich folgender Ablauf zum Aufbau einer Konzepthierarchie:

1) Für jedes Paar an vorhandenen Konzepten wir der kürzeste Pfad bestimmt und zur Konzepthierarchie hinzugefügt.

2) Für jedes Konzept werden alle streng aufsteigenden Pfade zum Wurzelkonzept der Konzepthierarchie hinzugefügt.

Die LCS werden bei Schritt 2) indirekt auch hinzugefügt da sie sich auf jeweils einem streng aufsteigenden Pfad zweier Konzepte befinden. Das hinzufügen aller Pfade von einem Knoten zur Wurzel ist wichtig, da nur dann alle Themenzugehörigkeiten gefunden werden können. Umgekehrt kann nur so für Konzepte ihr Status als Thema, nach den Ansätzen von Stoica et al., in der Konzepthierarchie korrekt bestimmt werden.

5.5 Konsolidierung der gewonnen Konzepthierarchien

Nachdem in den vorangegangenen Abschnitten Wissen aus den einzelnen Quellen extrahiert wurde und jeweils eine Konzepthierarchie für jede Wissensquelle aufgebaut wurde, soll nun analysiert werden wie die Konzepthierarchien konsolidiert werden können. Ziel ist eine einheitliche Hierarchie, die die Informationen der einzelnen Quellen vereinigt und dadurch eine einheitliche Schnittstelle für die Untersuchung der semantischen Beziehung der Entitäten in Anforderungen zur Verfügung stellt. Der Vorgang der Konsolidierung soll wie alle anderen Schritte auch voll automatisch funktionieren. In Abschnitt 5.5.1 wird betrachtet welche Probleme beim Konsolidieren von Konzepthierarchien gelöst werden müssen. In Abschnitt 5.5.2 wird daraufhin analysiert welche bestehenden Verfahren zur Konsolidierung von Wissensquellen existieren und inwiefern sie im Rahmen dieser Arbeit nützlich sein können. In Abschnitt 5.5.3 wird auf Basis bestehender Verfahren untersucht, wie die aufgebauten Konzepthierarchien tatsächlich konsolidiert werden können.

5.5.1 Problemanalyse für die Konsolidierung von Konzepthierarchien

Bevor Verfahren zur automatischen Konsolidierung von Konzepthierarchien betrachtet werden, soll in diesem Abschnitt zunächst untersucht werden, aus welchen Teilproblemen sich die Konsolidierung einer Konzepthierarchie zusammensetzt. In der Literatur werden die verwendeten Wissensquellen aufgrund ihrer Struktur meist als Ontologien bezeichnet. Daher wird meist der Begriff des Vereinigens von Ontologien (engl. ontolgy merging) (siehe Abschnitt 2.4.2) verwendet, wenn über die Konsolidierung von Konzepthierarchien gesprochen wird. Eine Disziplin die mit dem Vereinigen von Ontologien einhergeht ist die

Disziplin des Abbildens von Ontologien (engl. ontology mapping). Bei dieser geht es darum eine Abbildung zwischen den Konzepten der verschiedenen Ontologien herzustellen, die inhaltlich das gleiche meinen und in einer verbunden Ontologie nur noch durch einen Knoten dargestellt werden würden. Die Abbildung aller Konzepte A aus O_A auf die identischen Konzepte $B(= A)$ aus O_B dient also als Anleitung dafür, welche Konzepte identisch sind. Ein Abbildung zwischen den verschiedenen Arten von Relationen in den Wissensquellen wird bereits in Abschnitt 5.2 beim Entwurf eines Datenmodells definiert. Mit dieser Information und der Abbildung der Konzepte lässt sich ableiten welche Relationen identisch sind.

Um zwei Ontologien, die bereits hinsichtlich der Art des Datenmodells aufeinander abgestimmt wurden, zu einer zu vereinigen sind also folgende Schritte notwendig:

1) Berechnung der wahrscheinlichsten Abbildung aller Konzepte A aus O_A auf die identischen Konzepte $B(= A)$ aus O_B.

2) Entscheidung, ob es sich bei der wahrscheinlichsten Abbildung tatsächlich um ein identisches Konzept handelt, oder ob mindestens ein unter Umständen anderer Umfang im konkreten Anwendungsfall vernachlässigbar ist.

3) Kopieren aller Konzepte in eine neue Ontologie und zusammenführen der Konzepte die nach Schritt 2) identisch sind.

4) Kopieren aller Konzeptbeziehungen in die neue Ontologie. Quell- und Zielknoten einer Beziehung ergeben sich aus den Ursprungsreferenzen der zusammengeführten Knoten.

5) Entfernen aller Beziehungen, die den Eigenschaften der Konzepthierarchie (siehe Abschnitt 2.4.1) widersprechen. Reflexivität und Transitivität werden durch die Vereinigung nicht beeinflusst, aber es könnte Antisymmetrie entstanden sein. Dann muss mindestens eine der gegensätzlichen Kanten entfernt werden. Die Antisymmetrie könnte direkt durch eine gegensätzliche Verbindung, oder transitiv über mehrere Ebenen hinweg entstanden sein. Dann ist nicht offensichtlich bestimmbar welche Kanten die zu der gegensätzlichen Aussage beitragen entfernt werden müssten.

5.5.2 Bestehende Verfahren zur automatischen Konsolidierung von Konzepthierarchien

Zunächst wird der Stand der Technik bei der automatischen Konsolidierung von Konzepthierarchien untersucht. Die meisten Arbeiten verwenden den Begriff des Vereinigen von Ontologien (engl. ontolgy merging) (siehe Abschnitt 2.4.2) wenn sie über die Konsolidierung von Konzepthierarchien sprechen.

Einen Überblick über den Stand der Technik von 2001 gibt Klein in seiner Arbeit **Combining and relating ontologies: an analysis of problems and solutions** (siehe Abschnitt 4.2.1). Dort gibt er unter anderem einen Überblick über verschiedene, in Abbildung 5.6 dargestellte, Werkzeuge zum Vereinigen von Ontologien. Außerdem beschreibt Klein welche Teilaufgaben des Vereinigens automatisch (A), semi-automatisch (U) oder manuell (M) bewerkstelligt werden können. Dabei fällt auf, dass die meisten Werkzeuge für einen Großteil der Teilaufgaben nur manuelle Ansätze bieten. Es handelt sich also im Grunde um Werkzeuge die einen menschlichen Benutzer bei der Arbeit mit Ontologien unterstützen sollen, ohne dabei größere Aufgabenabschnitte automatisch ohne das Verständnis des Menschen lösen zu können. Die zwei Werkzeuge die dabei den höchsten Grad an Automatisierung vorweisen können sind PROMPT und Chimaera. Beide können laut Klein semi-automatisch den Umfang von Konzepten und die Verwendung verschiedener synonymer Worte zur Beschreibung von Konzepten auflösen. Außerdem können beide semi-automatisch eine Abbildung

Issues		SKC	Chim.	PROMPT	SHOE	OntoM.	Metamodel	OKBC	Layering
Language level mismatches	Syntax					M	M	M	M
	Representation					M	M	M	M
	Semantics					M	M		M
	Expressivity								
Ontology level mismatches	Paradigm					M			
	Concept description					M			
	Coverage of model								
	Scope of concepts	M	U	U	M	M			
	Synonyms	M	U	U	M	M			
	Homonyms	M				U			
	Encoding	M			M	M			
Practical problems	Finding alignments	U	U	U					
	Diagnosis of results		A	A		A			
	Repeatability	A		A		A			
Ontology versioning	Identification				M				
	Change tracking				M				
	Translation								

Abbildung 5.6: Vergleich verschiedener Werkzeuge zum Kombinieren von Ontologien, basierend auf ihrer Fähigkeit die genannten Probleme automatisch (A), semi-automatisch (U) oder manuell (M) auflösen zu können. Bildquelle: [Kle01]

mehrerer Ontologien aufeinander erzeugen und eine automatische Diagnose über mögliche Unstimmigkeiten in den resultierenden Ontologien durchführen. Mit semi-automatisch ist im Falle der beiden Werkzeuge gemeint, dass sie dem Nutzer Vorschläge zur Vereinigung von Ontologien liefern, die jedoch manuell überprüft werden müssen.

5.5.2.1 Bestehende Verfahren zur automatischen Abbildung von Konzepthierarchien

In Kapitel 4 wurde bereits ein kurzer Einblick in die Arbeit **Algorithm and Tool for Automated Ontology Merging and Alignment** gegeben. In dieser wird das Werkzeug PROMPT vorgestellt, welches mehrere Teilprobleme rund um die Konsolidierung von Ontologien behandelt. Unter anderem muss auch in PROMPT eine Abbildung zwischen den Konzepten der verschiedenen Ontologien bestimmt werden. Für PROMPT wird dabei auf die einfachste Methode des syntaktischen Vergleichs der Namen der Konzepte zurückgegriffen. Dabei werden alle Konzepte der abzubildenden Ontologien paarweise verglichen. Sind ihre Namen syntaktisch identisch oder mindestens sehr ähnlich, wird eine Abbildung der Konzepte vorgeschlagen, die dann vom menschlichen Bearbeiter manuell bestätigt werden muss. Diese Art von Vorgehen bei dem ein Vorschlag automatisch generiert wird, aber manuell über dessen Richtigkeit entschieden werden muss, wird semi-automatisch genannt. Zur Evaluation des Ansatzes wurden zwei Ontologien, die beide Problemlösungsmethoden modellieren vereinigt. Sie haben insgesamt 134 Elemente die zu 117 Elementen vereinigt wurden. Dabei folgten menschliche Experten 90% der Vorschläge des PROMPT Verfahrens. Insgesamt schlug PROMPT 74% der nötigen Operationen zum verbinden der Ontologien vor, wobei hier nicht nur das Abbilden sondern auch nachträgliche Auflösung von Konflikten, die erst im nächsten Abschnitt behandelt wird, mit eingerechnet ist. Die Autoren erwähnen jedoch, dass die Evaluationsergebnisse wesentlich durch die Auswahl der kleinen und ähnlichen Ontologien beeinflusst sein könnten.

Während das zunächst vorgestellte PROMPT-Verfahren nur den lokalen Kontext, also die jeweiligen Konzepte und ihre Eigenschaften betrachtet, stellen Noy und Musen in ihrem Artikel **The PROMPT Suite: Interactive Tools For Ontology Merging And** [NM03] mit ANCHOR-PROMPT eine erweiterte Variante vor, die auch die Umgebung in der Ontologie in den Prozess mit einbezieht. ANCHOR-PROMPT basiert, wie in Abbildung 5.7 dargestellt, auf der Annahme, dass für ein ähnliches in der Hierarchie höher gelegenes

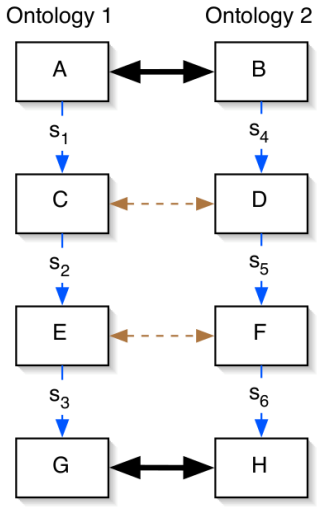


Abbildung 5.7: Der ANCHOR-PROMPT-Algorithmus basiert auf der Annahme, dass für die Paare (Anker) AB und GH mit Pfaden zwischen A und G sowie zwischen B und H, die Elemente auf den Pfaden ebenfalls Paare sind. Bildquelle: [NM03]

Konzept und ein zu diesem tiefer gelegenes Konzept mit entsprechendem Pfad zum Höheren, die Konzepte zwischen ihnen ebenfalls ähnlich sind. Dieser Zusammenhang wird formalisiert indem ein Ähnlichkeitswert für die Paare berechnet wird und dieser im Falle der entsprechenden Position in ANCHOR-PROMPT erhöht wird. Für eine Menge an gegebenen Ankern werden dann alle möglichen Pfade mit maximaler Länge L betrachtet und über den Ähnlichkeitswert gezählt wie oft zwei Konzepte an gleicher Stelle in einem solchen Pfad vorkommen. Noy und Musen weisen darauf hin, dass ANCHOR-PROMPT in ähnlich aufgebauten Ontologien vielversprechende Ergebnisse liefert, die dem Ansatz zugrundeliegende Annahme jedoch bei verschieden konstruierten Ontologien nicht standhält.

Ein anderes Verfahren das die Abbildung zwischen Ontologien mit maschinellem Lernen verbessern soll, ist GLUE. In der Arbeit **Ontology Matching: A Machine Learning Approach** [DMDH04] beschreiben Doan et al. GLUE als ebenfalls semi-automatisches Verfahren zum Finden von Paaren ähnlichster Konzepte zwischen Ontologien. GLUE berechnet dabei, unabhängig von einem konkreten Ähnlichkeitsmaß, die gemeinsame Verteilung zweier Konzepte A und B. Die gemeinsame Verteilung von A und B besteht aus $P(A, B)$, $P(A, \overline{B})$, $P(\overline{A}, B)$ und $P(\overline{A}, \overline{B})$ wobei $P(A, B)$ die Wahrscheinlichkeit dafür ist, dass eine Instanz sowohl zu Konzept A als auch zu Konzept B gehört. Auf Basis dieser gemeinsamen Verteilungen können dann verschiedene Ähnlichkeitsmaße wie die Jaccard-Ähnlichkeit verwendet werden, um einen konkreten Wert für die Ähnlichkeit zweier Konzepte zu erhalten. Die geteilten Wahrscheinlichkeiten lassen sich jedoch nicht unmittelbar bestimmen, da die Instanzen in den beiden Ontologien getrennt vorliegen und keine übergreifende Zugehörigkeit zu einem Konzept der jeweils anderen Ontologie bekannt ist. Die Idee hinter GLUE ist es deshalb einen Klassifizierer für A auf den Instanzen von A zu trainieren und daraufhin Instanzen aus B mit diesem zu klassifizieren und umgekehrt. Daraus ergeben sich die Werte für die geteilte Wahrscheinlichkeit der Konzepte. Der Klassifizierer besteht aus mehreren Unter-Klassifizierern, die bestimmte Eigenschaften wie den Namen und den Kontext der Konzepte zur Klassifikation verwenden und die gemäß einer manuell festgelegten Gewichtung zu einem Klassifikationsergebnis zusammen gerechnet werden. Wie auch bei anderen automatischen Abbildungsverfahren weißen Doan et al. in ihrer Arbeit darauf hin, dass

der Prozess der semantischen Abbildung hochgradig subjektiv und vom konkreten Ziel abhängig ist. Deshalb ist auch GLUE lediglich als semi-automatisches Verfahren, mit dem Ziel den Nutzer bei der manuellen Arbeit mit Ontologien zu unterstützen, konzipiert. Die im Vorangegangenen Verfahren sollen als Beispiele für eine ganze Reihe von Verfahren und Werkzeugen dienen, die eine Ähnliche Herangehensweise verfolgen.

Zusammenfassend lässt sich sagen, dass alle betrachteten Verfahren zum Abbilden von Ontologien semi-automatisch sind. Sie bieten verschiedene Ansätze um auf Basis von Syntax oder semantischen Beziehungen ein Maß für die Ähnlichkeit zweier Konzepte zu bestimmen. Diese Ansätze reichen von simplen syntaktischen Vergleichen zu komplexeren Vergleichen die Struktur und Kontext einbeziehen und die Kombinationen aus mehreren Verfahren. In allen Fällen verweisen die Urheber der vorgestellten Verfahren allerdings darauf, dass das Abbilden von Ontologien im Kern subjektiv ist und die vorgestellten Verfahren deshalb nur als semi-automatische Verfahren sinnvoll einsetzbar sind. Eine Herausforderung ist dabei zu entscheiden ab welchem Ähnlichkeitswert Konzepte als identisch betrachtet werden können und ob ein solcher Schwellwert überhaupt zuverlässig bestimmt werden kann. In den Evaluationen der verschiedenen Verfahren haben diese grundsätzlich vielversprechende Ergebnisse geliefert. Allerdings wurden dabei Ontologien verwendet die mit wenigen hundert Konzepten sehr klein waren und die auf die gleiche Art und Weise konstruiert und deshalb grundsätzlich vom Aufbau her sehr ähnlich waren.

5.5.2.2 Bestehende Verfahren zur automatischen Vereinigung von Konzepthierarchien mit vorhandener Abbildung

Nachdem eine Abbildung der Konzepte aus einer Ontologie auf die Konzepte der Anderen bestimmt wurde, können zwei Ontologien zu einer vereinigt werden. Dies funktioniert zum Beispiel beim im vorhergehenden Abschnitt beschriebenen PROMPT-Verfahren, das neben der Komponente zum Abbilden auch eine zum Vereinigen enthält, wie im folgenden beschrieben. Während der Vereinigung werden Konzepte aus den Quell-Ontologien nach O_M kopiert und ihr Ursprungskonzept referenziert. Dort können Konzepte dann zu einem vereinigt werden und inhaltliche Änderung zur Auflösung von Widersprüchen vorgenommen werden. Kandidaten für eine Vereinigung sind dabei die Konzepte die aufeinander abgebildet wurden. Ob die beiden Konzepte tatsächlich vereinigt werden ist nach Noy und Musen subjektiv und hängt deshalb von einer manuellen Entscheidung des menschlichen Nutzer ab. Beim Vereinigen von Konzepten wird für die Konzepte A und B aus verschiedenen Ontologien ein Konzept M in der Ziel-Ontologie O_M erzeugt. Für jedes Konzept C, das Über- oder Unterkonzept von A oder B ist und das in O_M existiert, ist C_M ebenfalls Über- oder Unterkonzept von M.

Durch das vereinigen von Konzepten, können Konflikte in der Konzepthierarchie entstanden sein. Im Rahmen von PROMPT wurden vier solcher Konflikte identifiziert. Tritt einer der Konflikte auf, wird der Nutzer des PROMPT-Werkzeuges darauf hin gewiesen. Anders als beim Abbilden, wo nur eine Bestätigung durch den Nutzer nötig war um den Vorschlag zu vervollständigen muss hier der Nutzer tatsächlich selbst den Konflikt auflösen. Die vier identifizierten Konflikte sind Konflikte bei der Benennung von Konzepten, schwebende Beziehungen ohne Zielknoten, Redundanz in der Hierarchie und Konflikte bei Attributen von Konzepten. Im Rahmen dieser Arbeit ist dabei nur der dritte genannte Konflikt relevant. Konflikte bei der Benennung treten nicht auf, da der Name im entwickelten Datenmodell nicht eindeutig sein muss, sondern nur der Identifikator eines Konzeptes. Mehrfach auftretende Namen sind bei den großen Wissensquellen aufgrund von Homonymen sogar wahrscheinlich. Schwebende Beziehungen ohne Zielknoten treten bei den aufgebauten Konzepthierarchien zwangsläufig auf, da nicht alle Konzepte aus den ursprünglichen Wissensquellen übernommen wurden. Attribute bei deren Vereinigung es zu Problemen kommen könnte existieren im entwickelten Datenmodell nicht. Der dritte Fall zeigt dafür

eine entscheidende Schwierigkeit bei der Vereinigung von Konzepten. Immer dann wenn die Pfade zu umliegenden Konzepten der zwei Quellkonzepte eines vereinigten Konzeptes sich unterscheiden, entstehen redundante Verbindungen zu umliegenden Konzepten. Diese müssen dann manuell aufgelöst werden, indem einer der Pfade entfernt wird.

5.5.3 Anwendbarkeit des Standes der Technik

Nachdem ein Überblick über den Stand der Technik in den Gebieten Abbilden und Vereinigen von Ontologien gegeben wurde, soll nun untersucht werden, wie die Ergebnisse sich auf den Entwurf einer Lösung zum Aufbau einer Konzepthierarchie aus beliebigen Wissensquellen auswirken. Dafür wird zunächst in Abschnitt 5.5.3.1 betrachtet wie die im Rahmen dieser Arbeit aufgebauten Konzepthierarchien aufeinander abgebildet werden können. Danach wird in Abschnitt 5.5.3.2 untersucht wie die aufeinander abgebildeten Konzepthierarchien zu einer vereinigt werden können.

5.5.3.1 Abbilden von Konzepthierarchien

Die einfachste Möglichkeit zum Abbilden von Konzepthierarchien ist die Abbildung anhand syntaktischer Überschneidungen zwischen den Namen und Beschreibungen zweier Konzepte (siehe Abschnitt 5.5.2.1). Dieser Ansatz entspricht dem Lesk-basierten Ansatz aus Abschnitt 5.3 wobei für die Namen und Beschreibungen eines Konzeptes das am besten passendste Konzept aus der jeweils anderen Konzepthierarchie gesucht wird. Im Grund liegt die gleiche Art von Problem vor. Allerdings gibt es in den aufgebauten Konzepthierarchien vermutlich weniger Konzepte mit syntaktisch gleichen Namen, da nur ausschnitte der gesamten Wissensquellen hinzugefügt werden und homonyme Begriffe Konzepten in unterschiedlicher Bereiche der Wissensquellen zugrunde liegen. Um die Ergebnisse weiter zu verbessern, könnten Informationen über umliegende Konzepte in den Konzepthierarchien in den Abbildungsprozess eingebunden werden. Ein Beispiel für einen solchen strukturbasierten Ansatz ist die vorgestellte ANCHOR-PROMPT-Erweiterung. Diese scheint allerdings für den Anwenndungsfall dieser Arbeit nicht sinnvoll. Wie bereits beschrieben weißen die Urheber des Verfahrens darauf hin, dass es nur für ähnlich aufgebaute Ontologien funktioniert. Für Wikidata und WordNet ist dieser Ansatz deshalb nicht sinnvoll, denn es liegen meist grundlegende Unterschiede in der Art der Modellierung vor. Ein andere Möglichkeit Strukturwissen in den Prozess einzubeziehen ist der bei der Bestimmung von Einstiegskonzepten verwendete Ansatz die Namen der benachbarten Konzepte einzubeziehen. Sind die Namen zweier Konzepte gleich und die deren Nachbarn auch, dann ist die Wahrscheinlichkeit vermutlich hoch, dass es sich um das gleiche Konzept bei gleicher oder ähnlicher Granularität handelt. Andere Verfahren wie das vorgestellte GLUE-Verfahren setzten auf eine ausreichende Menge von Instanzen um die Beziehungen zwischen zwei Konzepten auf Basis derer zu bestimmen. Das aus Wikipedia-Artikeln abgeleitete Wikidata enthält eine vergleichsweise große Menge an Instanzen und wäre somit ein interessanter Kandidat für derartige Ansätze. WordNet enthält dagegen kaum Instanzen und wenn dann nicht in der technischen Domäne, sodass eine Abbildung zwischen den beiden Wissensquellen auf Basis von Instanzen nicht möglich ist.

Eine Bestimmung von Abbildungen scheint also zumindest mit primitiven Methoden umsetzbar zu sein. Dies gilt zumindest dann wenn man auf die Korrektheit der Abbildung nur zu einen gewissen grad angewiesen ist. Was mögliche Fehler in der Abbildung für deren Verwendbarkeit in der Vereinigung bedeuten wird im nächsten Abschnitt behandelt. Grundlegend scheinen folgende Regeln bei der Abbildung zu mindest teilweise zu gelten:

1) Zwei Konzepte die mindestens eine Überschneidung bei den zugrundeliegenden Namen haben beschreiben vermutlich ein Ähnliches Konzept.

2) Zwei Konzepte für die 1) gilt und die zusätzlich jeweils mindestens ein Konzept in ihrer Nachbarschafthaben für die zusammen ebenfalls 1) gilt haben eine erhöhte Wahrscheinlichkeit in ähnlicher Granularität vorzuliegen.

Aus diesen Regeln lässt sich folgender Ablauf für die Erzeugung einer Abbildung zwischen zwei Konzepthierarchien ableiten. Für jedes Konzept $K1_A$ der Konzepthierarchie $K1$ für das genau ein Konzept $K2_A$ in Konzepthierarchie $K2$ mit mindestens einem identischen oder ähnlichen Namen existiert, wird eine Referenz von $K1_A$ auf $K2_A$ und umgekehrt gespeichert. Daraufhin wird für jede Referenz gezählt wie viele identische oder ähnliche Begriffe in den direkt benachbarten Konzepten vorkommen und dieser Wert als Attribut der Referenz hinzugefügt. Erste Versuche haben gezeigt das dieser Wert für Worte die für eine Abbildung in Frage kommen mindestens eins ist.

5.5.3.2 Vereinigen abgebildeter Konzepthierarchien

Bei der Vereinigung sind zwei grundlegende Probleme zu lösen. Zum einen muss entschieden werden welche Konzepte zu einem vereinigt werden sollen. Diese Entscheidung wird indirekt schon bei der Abbildung getroffen. Allerdings wird dort oft nur der am besten geeignete Kandidat für eine Abbildung gesucht. Je nach Anwendungsfall reicht das als Qualität für die Abbildung aus. Sobald zwei Konzepthierarchien jedoch vereinigt werden sollen, stellt sich für jede Abbildung die Frage ob sie tatsächlich eine Anweisung zur Vereinigung darstellt. Es geht also bei der Vereinigung im ersten Schritt darum für aufeinander abgebildete Konzepte zu entscheiden ob eine Vereinigung sinnvoll ist oder nicht. Diese Entscheidung lässt sich wohl nicht ohne weiteres treffen. Folgt man der Ansicht von Noy und Musen, oder der von Klein ist sie sogar subjektiv, denn es lässt sich in vielen Fällen nicht sagen ob eine Vereinigung richtig oder falsch wäre. Stattdessen geht es eher um die persönlichen Ansichten des Erstellers einer Wissensquelle und den Anwendungsfall. Ein Anwendungsfall lässt sich im Rahmen dieser Arbeit nur schwer festlegen. Zum einen wurden zwar in Abschnitt 5.4 zwei mögliche Anwendungsfälle vorgestellt. Zum anderen ist es aber Ziel der Arbeit eine einheitliche Konzepthierarchie ohne Beschränkung auf eine bestimmte Aufgabe zu erstellen. Auch die Ansichten des Nutzers lassen sich bei einem automatisierten Verfahren nur schwer als Entscheidungsgrundlage für die Abbildung heranziehen.

Das zweite Kernproblem beim Vereinigen von Konzepthierarchien ist es, nach der Vereinigung von Konzepten die entstandenen Konflikte aufzulösen. Das Hauptbeispiel bei der Auflösung von Konflikten waren redundante Pfade. Immer dann wenn die Pfade zu umliegenden Konzepten der zwei Quellkonzepte eines vereinigten Konzeptes sich unterscheiden, entstehen redundante Verbindungen zu umliegenden Konzepten. Ein Beispiel für diesen Fall sind die Pfad des Konzeptes „system“ zum Konzept „entity“ in WordNet und Wikidata:

$$\begin{aligned} &Wikidata : system- > structure- > entity \\ &WordNet : system- > group- > abstraction- > entity \end{aligned} \tag{5.9}$$

Das Konzept „entity“ aus Wikidata und WordNet ist die Wurzel der jeweiligen Wissensquelle und damit automatisch das identische Konzept. Entscheidet man sich nun dazu die beiden Konzepte „system“ zu einem zu vereinigen, enthält die neue vereinigte Konzepthierarchie nun beide dargestellten Pfade zu „entity“. Diese Tatsache ist ein ungewollter Konflikt, denn das Resultat sind im Grunde wieder zwei getrennte Vorstellungen der Beziehungen von Konzepten. Im Grunde wäre das speichern zweier redundanter Pfade für ein vereinigtes Konzept genauso als ob man die Konzepte gleich in ihren Ausgangshierarchien lässt und nur eine Abbildung hinterlegt. Gleichzeitig scheint es keinen sinnvollen Anhaltspunkt zu geben um sich für einen der Pfade zu entscheiden. Auch Klein greift

diesen Konflikt in seiner Arbeit auf (siehe Kapitel 4). Er und weitere Arbeiten argumentieren, dass es in dieser Situation keinen universell objektiv besseren oder richtigeren Pfad gibt, sondern die Entscheidung davon abhängt welcher Pfad eher der Vorstellung des Nutzers der Ontologie entspricht. Letztendlich sind Ontologien Modelle der Realität und die Anzahl der Möglichkeiten einen bestimmten Zusammenhang zu modellieren endlos. Tatsächlich ist das gezeigte Beispiel für die unterschiedlichen Pfade von „system“ zu „entity“ kein Randfall sondern in WordNet und Wikidata eher häufig.

Zusammenfassen gilt, dass es keinen universellen Weg gibt um eine allgemeingültige Konzepthierarchie aus mehreren voneinander getrennten zu vereinigen. Es ist denkbar das verschiedene Strategien wie zum Beispiel die Entfernung des kürzeren oder des längeren Pfades bei Redundanzen, eine Art von Vereinigung technisch gesehen möglich machen. Da aber keine Aussage darüber welches Ergebnis besser oder schlechter ist gemacht werden kann, können solche Ansätze nur im Kombination mit einer klaren Aufgabenstellung und damit einhergehend mit einem messbaren Mehrwert des Vereinigungsergebnisses vollzogen werden. Auch für die anderen Probleme der Vereinigung wie der Frage zu welchem Grad zwei aufeinander abgebildete Konzepte sich entsprechen könnten Strategien wie Schwellwerte experimentell untersucht werden. Da aber zum Zeitpunkt dieser Arbeit noch kein klar definierter Anwendungsfall mit entsprechend messbaren Mehrwert vorliegt, kann keine sinnvolle Vereinigung mehrerer Konzepthierarchien vollzogen werden.

6 Implementierung

In diesem Kapitel wird die Implementierung des in Kapitel 5 entworfenen Ansatzes beschrieben. Wie in Kapitel 3 beschrieben, ist INDIRECT als System aus mehreren unabhängigen Agenten entwickelt. Im Zentrum dieser Agenten steht ein Graph-Speicher auf dessen Daten die Agenten ihre Operationen ausführen falls sie die Voraussetzung für deren Ausführung auf dem Graph-Speicher vorfinden.

Auch der im Rahmen dieser Arbeit entwickelte Ansatz soll dementsprechend als einer oder mehrere Agenten in INDIRECT integriert werden. Dabei soll zunächst auf die Verarbeitungsergebnisse des EntiyRecognizers (siehe Abschnitt 3.2) gewartet werden. Dieser stellt alle Entitäten die aus den untersuchten Anforderungen extrahiert wurden als Knoten im Graph-Speicher bereit. Für diese Knoten werden dann, wie im Entwurf beschrieben, die Einstiegskonzepte gesucht, Konzepthierarchien für jede Wissensquelle aufgebaut und diese dann aufeinander abgebildet. Diese drei unterschiedlichen Schritte könnten durch drei verschiedene Agenten umgesetzt werden. Im Rahmen dieser Arbeit wurde jedoch die Entscheidung gefällt diese zunächst in einem zentralen Agenten zu implementieren. Grund dafür ist das bei konzeptionellen Änderungen während des Entwicklungsprozesses durch die Trennung in verschiedene Agenten zusätzlicher Aufwand entstanden wäre. Stattdessen werden die drei Schritte in drei Klassen aufgeteilt, sodass sie später, sobald ein besseres Verständnis für den gesamten Entwurf der anderen Agenten vorliegt, einfach in mehrere Agenten aufgeteilt werden können. Im folgenden wird der Aufbau dieses Agenten beschrieben.

6.1 Aufbau des Agenten

Der entwickelte Agent basiert auf zwei Schnittstellen und drei Grundoperationen. Die grundlegende Funktionsweise ist, dass zunächst Entitäten aus dem **IGraph** in den **ConceptGraph** geladen werden. Dort werden dann die entsprechenden Einstiegskonzepte durch den **EntryConceptBuilder** gesucht und zum **ConceptGraph** hinzugefügt. Für die gefundenen Einstiegskonzepte werden dann durch den **ConceptCollector** die Konzepte zum Aufbau von Konzepthierarchien für die jeweiligen Wissensquellen eingesammelt und zum **ConceptGraph** hinzugefügt. Der beschriebene Ablauf ist in Abbildung 6.1 dargestellt.

6.1.1 ConceptGraph

Die **ConceptGraph**-Klasse beinhaltet die aufgebaute Konzepthierarchie. Sie läd bei Initialisierung alle vorhanden Entitäten des **IGraph**. Die weiteren Verarbeitungsschritte rund

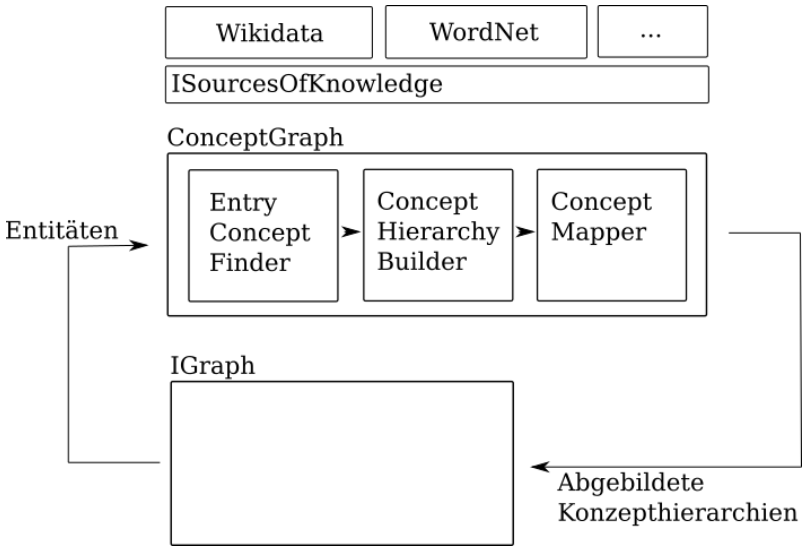


Abbildung 6.1: Schematischer Überblick über den entwickelten Konzeptualisierungsagenten.

um den Aufbau und das Abbilden von Konzepthierarchien finden auf dem **ConceptGraph** statt. Nachdem alle Verarbeitungsschritte beendet sind, überträgt der **ConceptGraph** die aufgebaute Konzepthierarchie in den **IGraph**.

6.1.2 Schnittstelle für beliebigen Wissensquellen

Um mit beliebigen Wissensquellen unabhängig von deren Implementierung und Format arbeiten zu können, wir eine Schnittstelle entwickelt. Die Wissensquellen stellen über diese vier Methoden bereit. Diese dienen dazu ein Konzepte über dessen Identifikation zu bekommen, alle Konzepte mit einem bestimmten Begriff als zugrundliegendem Namen zu bekommen und Pfade zwischen den Konzepten in Wissensquellen zu finden. Dabei kann der kürzeste Pfad zwischen zwei Konzepten oder alle Pfade eines Konzeptes zur Wurzeln der Wissensquelle abzufragen. Die konkrete Implementierung der Schnittstelle für die im Rahmen der Arbeit verwendeten Wissensquellen wird in Abschnitt 6.2 beschrieben.

6.1.3 EntryConceptFinder

Die **EntryConceptFinder**-Klasse stellt die Funktionalität bereit um für die Entitäten im **ConceptGraph** die entsprechenden Konzepte in den Wissensquellen zu finden.

6.1.4 ConceptHierarchyBuilder

Die **ConceptHierarchyBuilder**-Klasse stellt die Funktionalitäten bereit um mit den im **ConceptGraph** gesammelten Einstiegskonzepten für jede Wissensquelle eine Konzepthierarchie aufzubauen. Dazu wird das in Abschnitt 5.4 beschriebene Verfahren genutzt.

6.1.5 ConceptMapper

Die **ConceptMapper**-Klasse stellt die Funktionalität bereit um Abbildungen zwischen den im **ConceptGraph** aufgebauten Konzepthierarchien zu finden. Er folgt dabei dem Entwurf aus Abschnitt 5.5.3.1.

6.2 Implementierung der Schnittstelle für Wissensquellen

Die in Abschnitt 6.1.2 beschriebene Schnittstelle muss im Rahmen der Implementierung für die beiden Wissensquellen WordNet und Wikidata umgesetzt werden. In diesem Abschnitt wird beschrieben wie die Implementierung für WordNet und Wikidata funktioniert.

6.2.1 WordNet

Zur Einbeziehung von Konzepten aus WordNet wird die „Extended Java WordNet Library“ (extJWNL), eine Java Bibliothek für den Zugriff auf eine lokale Kopie der WordNet Datenbank verwendet. Mit dieser kann eine Instanz von WordNet geladen und verschiedene Anfragen an diese gestellt werden. Die Umsetzung des Zugriffs auf Konzepte basiert dabei auf der Suche nach Synsets auf Basis ihrer Identifikation oder auf Basis ihres Namen. Diese beiden Funktionen werden durch die extJWNL bereitgestellt. Zur Suche nach kürzesten Pfaden in WordNet wird eine Breitensuche implementiert. Zur Suche nach Pfaden von einem Knoten zur Wurzel von WordNet wird ein Ablauf implementiert, der alle Hyperonym Beziehungen verfolgt bis keine mehr gefunden werden.

6.2.2 Wikidata

Der Zugriff auf Wikidata wird über den SPARQL-Endpunkt http://query.wikidata.org realisiert. Alternativ existieren auch Versionen von Wikidata, die für die lokale Entwicklung heruntergeladen werden können. Allerdings sind diese inzwischen mehrere hundert Gigabyte, sodass deren Verwendung auf einem Entwicklungsgerät nicht praktikabel ist. Außerdem wird durch die Verwendung des Online-Endpunktes immer die aktuellste Version von Wikidata geladen. Für den produktiven Einsatz sollte jedoch der Zugriff auf Wikidata auf eine lokale Version umgestellt werden.

7 Evaluation

Im Rahmen dieser Arbeit wurde ein Verfahren entwickelt, dass die Entitäten die in Anforderungen vorkommen mit entsprechenden Konzepten in Wissensquellen verknüpft und aus den gewonnen Informationen eine konsolidierte Konzepthierarchie erstellt. Der Ansatz dabei mehrere Wissensquellen zu kombinieren basiert auf der Annahme, dass diese sich ergänzende Informationen enthalten. In Abschnitt 7.2 wird deshalb untersucht wie gut die Abdeckung der Wissensquellen WordNet und Wikidata für Entitäten aus Anforderungsdokumenten ist. Um die Entitäten mit den passenden Konzepten in den Wissensquellen zu verbinden, wurde ein Verfahren entwickelt um die passenden Konzepte zu finden. Dies basiert im Grunde auf dem Lesk-Algorithmus und soll immer dann zum Einsatz kommen wenn kein besserer Wort-Sinn-Disambiguierer für eine spezifische Wissensquelle verfügbar ist. Im Abschnitt 7.3 wird untersucht wie gut dieses allgemeine Verfahren funktioniert. Nachdem die passenden Einstiegskonzepte ausgewählt wurden, verbindet das entwickelte Verfahren diese in einer Konzepthierarchie und versucht für Konzepte die einer Wissensquelle W unbekannt sind mithilfe der anderen Wissensquellen einen Einstiegspunkt in W zu finden.

7.1 Evaluationsdatensatz

Für die Evaluation der Verwendung mehrerer Wissensquellen (siehe Abschnitt 7.2) und die Evaluation der Bestimmung der Einstiegskonzepte (siehe Abschnitt 7.3) wird ein Datensatz benötigt, der für eine Menge von Entitäten aussagt, ob sie in den Wissensquellen WordNet und Wikidata repräsentiert sind und wenn ja welches Konzept sie im Kontext eines konkreten Beispiels beschreiben. Die Begriffe des Datensatzes sollten aus Anforderungsdokumenten stammen, um für das entsprechend spezielle Vokabular dieser zu evaluieren. Damit ließe sich zum einen Untersuchen ob die beiden Wissensquellen für Begriffe die in Anforderungen verwendet werden die passenden Konzepte enthalten. Zum anderen ließe sich untersuchen wie gut der in dieser Arbeit beschriebene Ansatz zur Disambiguierung zur Auswahl des passenden Konzeptes funktioniert. Als Basis für einen solchen Datensatz dient im Rahmen dieser Evaluation der PURE-Datensatz [FSG17], der 79 verschiedene Anforderungsdokumente enthält. Aus ihm werden 33 Anforderungen in Form von natürlichsprachlichen Sätzen zufällig aus sechs verschiedenen Anforderungsdokumenten ausgewählt. Anforderungen werden dabei über eine zufällige Seitenzahl und eine zufällige Position auf der gezogenen Seite gewählt. Dieser Vorgang wird wiederholt bis eine für den Anwendungsfall passende Anforderung gezogen wird. Nicht passend sind Anforderungen die nicht aus

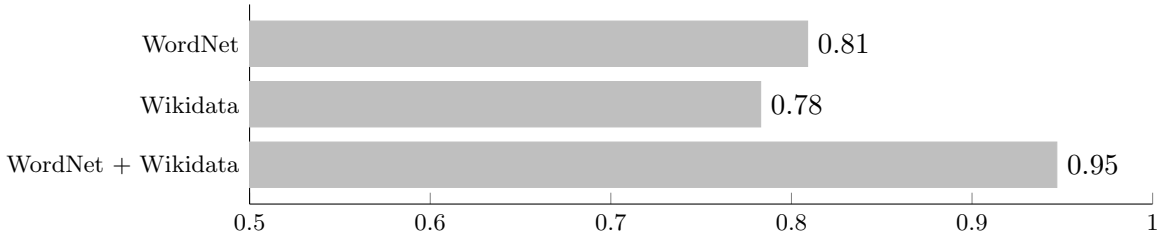


Abbildung 7.1: Die Abbildung zeigt die Abdeckung von 152 Entitäten für die Wissensquellen WordNet und Wikidata.

einem oder mehreren natürlichsprachlichen Sätzen aufgebaut sind. Für die ausgewählten Anforderungen werden dann die Entitäten ausgewählt und in einem Datensatz zusammengetragen. Für jede Entität wird dann manuell geprüft, ob ein passendes Konzept in WordNet und Wikidata gibt und die entsprechende Identifikationsnummer des passenden Konzeptes in den Datensatz übernommen. Gibt es in beiden Wissensquellen kein passendes Konzept, wird der Name der Entität nach den in Abschnitt 5.3 beschriebenen Regeln reduziert. Daraufhin wird geprüft ob es ein passendes Konzept für den reduzierten Namen gibt. Ist dies der Fall werden die entsprechenden Identifikationsnummern in eine dafür vorgesehene Spalte eingefügt. Die Tabelle in Abschnitt B des Anhangs gibt den Inhalt und die Struktur des Datensatzes wieder. Dabei steht in der ersten Spalte die Anforderung, in der zweiten Spalte der vollständige Name der Entität, in der dritten Spalte die mögliche WordNet-ID für den vollständigen Namen der Entität, in der vierten Spalte die mögliche Wikidata-ID für den vollständigen Namen der Entität, in der fünften Spalte die am wenigsten reduzierte Form des Namen der Entität für den mindestens eine der Wissensquellen ein Konzept hat, in der sechsten die mögliche WordNet-ID für diesen reduzierten Namen und in der siebten Spalte die Wikidata-ID für den diesen.

7.2 Kombination von Wissensquellen

Eine wichtige Annahme für diese Arbeit ist, dass sich die verwendeten Wissensquellen ergänzen, sodass ein Mehrwert durch deren Kombination geschaffen wird. Um diese Annahme für WordNet und Wikidata zu prüfen, wird für die Entitäten des in Abschnitt 7.1 erstellten Datensatzes geprüft ob für sie ein entsprechendes Konzept in WordNet, Wikidata oder in beiden existiert. Der Anteil einer Wissensquelle an Entitäten für die sie ein passendes Konzept besitzt gegenüber allen Entitäten wird als Abdeckung der Wissensquelle bezeichnet.

Abbildung 7.1 zeigt die Ergebnisse der Berechnung der Abdeckung. Dabei ist im Datensatz für 82% der Entitäten ein passendes Konzept in WordNet und für 78% der Entitäten ein passendes Konzept in Wikidata vorhanden. Für 95% der Entitäten ist entweder in WordNet oder in Wikidata ein passendes Konzept vorhanden.

Weitere Aussagen lassen sich durch die Betrachtung, für welche Entitäten jeweils Konzepte bekannt sind und für welche nicht, treffen. Dabei lassen sich zwei Gründe identifizieren:

(1) Die erste Situation ist die unübliche Verwendung eines Begriffes. In einem der Anforderungsdokumente wird der Begriff „screen“ mehrfach zu Bezeichnung einer Seite in einer Anzeige von Suchergebnissen bezeichnet. WordNet und Wikidata kennen den Begriff jedoch im Kontext von Computern lediglich als ein Wort für das Konzept des Monitors, also einem elektrischen Gerät. Auch wenn man den Begriff in den gängigen Internetsuchmaschinen sucht, gibt es fast ausschließlich Ergebnisse die sich auf die Bedeutung als Monitor beziehen. Es scheint aber eine Verwendung des Begriffes in der Domäne der Gestaltung

von graphischen Benutzeroberflächen zu geben. Dabei wird der Begriff „screen“ als irgendetwas das auf einem Monitor in einem bestimmten Moment angezeigt werden würde bezeichnet. Eine solche spezielle Verwendung eines Begriffes ist in Wikidata und WordNet möglicherweise nicht abgebildet.

(2) Die zweite Situation ist die Verwendung spezieller Instanzen, wie zum Beispiel der Abkürzung eines seltenen Dateiformates. Auch wenn Wikidata eine hohe Abdeckung für spezielle Instanzen wie bestimmte seltene Dateiformate hat, gibt es vereinzelte Instanzen die nicht vertreten sind. Wikidata kennt 3732 verschiedene Dateiformate, aber nicht das im Evaluationsdatensatz erwähnte NBI-Format.

Des Weiteren kann durch die Betrachtung der Entitäten, für die jeweils nur eine der beiden Wissensquellen ein passendes Konzept enthält, untersucht werden wo die Lücken bei den einzelnen Wissensquellen liegen. Zunächst werden die Entitäten betrachtet, für die nur WordNet ein passendes Konzept hat. Dabei fällt auf das WordNets Stärke bei allgemeinen Begriffen liegt. Es existieren in WordNet passende Konzepte für den „actor“, im Sinne einer handelnden Person, den „sender“, im Sinne einer Person die etwas verschickt und den „entry“, im Sinne eines Eintrags in einer Liste. Obwohl Wikidata über 450 mal soviele Konzepte kennt wie WordNet sind solche allgemeinen Konzepte dort nicht modelliert. So existiert beispielsweise der „dictionary entry“ im Sinne eines Wörterbucheintrages und trotzdem fehlt das Konzept des „entry“ im Sinne eines Eintrages in eine Sammlung oder Liste allgemein. Wikidata hat dafür eine große Menge an Konzepten die eine Instanz eines allgemeinen Konzeptes sind. In Abschnitt 5.1 wurde bereits beschrieben, dass Wikidata aus Wikipedia-Artikeln abgeleitet wurde. Es scheint, dass Wikipedia Artikel vor allem für Konkrete Dinge wie ein bestimmtes Dateiformat oder ein bestimmtes Rahmenwerk geschrieben wurden. Deswegen hat in Wikidata das Konzept „file format“ auch 3732 verschiedene Instanzen und das Konzept „software library“ 608. Im Datensatz schlägt Wikidata WordNet daher bei der Erkennung von Konzepten wie „xml“, „pdi“ und „pdf“. Auch eine allgemeine Nähe zur Software-Domäne lässt sich in Wikidata beobachten. Viele extrem spezifische Rahmenwerke und Konzepte einzelner Programmiersprachen sind in Wikidata abgebildet. Im Evaluationsdatensatz zeigt sich das durch die Erkennung von Konzepten wie „button“, „error message“, „session“ und „socket“.

Aufgrund dieser Nähe zur Softwaredomäne wurde zunächst vermutet, das Wikidata eine höhere Abdeckung für Konzepte aus Anforderungsbeschreibungen haben müsste. Tatsächlich ist dies aber wie in Abbildung 7.1 zu sehen nicht der Fall. Der Grund dafür ist, dass die in den Anforderungen des Evaluationsdatensatzes weniger technischer sind als zunächst vermutet. Die Sprache die zur Beschreibung von Funktionen in den evaluierten Anforderungen verwendet wird ist stärker an die alltägliche Sprache angelehnt als vermutet.

Insgesamt wird festgestellt, dass die Kombination von WordNet und Wikidata einen signifikanten Effekt auf die Abdeckungsrate für Konzepte in Anforderungen bringt. Die beiden Wissensquellen ergänzen sich dadurch, dass WordNet eine solide Menge an allgemeinen Konzepten und Wikidata eine große Anzahl von konkreten Instanzen dieser allgemeinen Konzepte mitbringt sehr gut.

7.3 Bestimmung der Einstiegskonzepte

Um beliebige Wissensquellen einbinden zu können, auch wenn kein geeignetes trainiertes Modell zur Bestimmung des passenden Konzeptes vorliegt, wird im Rahmen der Arbeit ein auf dem Lesk-Algorithmus basierendes Verfahren zur Bestimmung der Einstiegskonzepte vorgeschlagen. Zur Evaluation der Tauglichkeit dieses Ansatzes wird für die Entitäten des erstellten Evaluationsdatensatzes (siehe Abschnitt 7.1) jeweils das passende Konzept in WordNet und Wikidata mit diesem Verfahren bestimmt. Anhand der im Evaluationsdatensatz markierten Musterlösungen für die passenden Konzepte, wird dann die Präzision

Tabelle 7.1: Stand der Technik bei der Wort-Sinn-Disambiguierung für Nomen auf den fünf Datensätzen Senseval 2 und 3 und SemEval '07, '13, '15.

Verfahren	Wissensquelle	F1
MFS-Baseline	WordNet	67.6
$Lesk_{ext}$	WordNet	54.1
$Babelfy$	WordNet	68.6
$Lesk_{ext}$+emb	WordNet	69.8
GlossBERT(Sent-CLS-WS)	WordNet	79.8
SemCor+WNGC, hypernyms (ensemble)	WordNet	81.4

Tabelle 7.2: Ergebnisse des auf dem Lesk-Algorithmus basierenden Verfahrens auf dem erstellten Evaluationsdatensatz.

Verfahren	Wissensquelle	Präzision	Ausbeute	F1
$1_{st}sense - Baseline$	WordNet	61.5	69.8	65.4
$1_{st}sense - Baseline$	Wikidata	46.8	60.4	52.7
$Lesk$	WordNet	39.4	27.5	32.4
$Lesk$	Wikidata	34.8	21.5	26.6
$1_{st}sense + Lesk$	WordNet	45.6	51.7	48.4
$1_{st}sense + Lesk$	Wikidata	40.3	52.0	45.5

(engl. precision), Ausbeute (engl. recall) und das F_1-Maß für die jeweilige Wissensquelle mit folgenden Formeln bestimmt:

$$\begin{aligned} \text{Präzision} &= \frac{richtig\ positiv}{richtig\ positiv + falsch\ positiv} \\ &= \frac{korrekter\ Wortsinn}{nicht\ leere\ Spalten\ nach\ Verfahren} \end{aligned} \tag{7.1}$$

$$\begin{aligned} \text{Ausbeute} &= \frac{richtig\ positiv}{richtig\ positiv + falsch\ negativ} \\ &= \frac{korrekter\ Wortsinn}{nicht\ leere\ Spalten\ nach\ Datensatz} \end{aligned} \tag{7.2}$$

$$F_1 = 2 * \frac{\text{Präzision} * \text{Ausbeute}}{\text{Präzision} + \text{Ausbeute}} \tag{7.3}$$

Als Vergleich wurden zunächst Werte betrachtet, die in anderen Arbeiten auf den fünf Senseval Datensätzen im Rahmen von [RCCN17] bei der Wort-Sinn-Disambiguierung erzielt wurden. Typischer Vergleichswert ist dabei die Most-Frequent-Sense-Baseline (MFS-Basline) beziehungsweise $1_{st}sense - Baseline$. In WordNet werden bei der Suche nach einem Konzept die möglichen Ergebnisse nach der Auftrittshäufigkeit in dem WordNet zugrundeliegenden Korpus geordnet. Die häufigste Bedeutung steht dabei an erster Stelle der Suchtreffer. Solange Texte in alltäglicher Sprache verfasst sind und sich dadurch die Verteilung von Wortbedeutungen an der des Korpus von WordNet orientiert, ist die MFS-Basline häufig die tatsächlich gesuchte Wortbedeutung. Die MFS-Basline ist in der Disambiguierung eine vergleichsweise starke Baseline, die nur von wenigen komplexen Verfahren geschlagen wird. In Tabelle 7.1 sind die F_1-Werte für Disambiguierung auf einem

bekannten Vergleichsdatensatz dargestellt. Nur eine stark erweiterte Version des Lesk-Algorithmus kann die MFS-Basline schlagen. Die Ergebnisse der Evaluation des auf Lesk basierenden Ansatzes sind in Tabelle 7.2 dargestellt. Dabei wird für die MFS-Baseline in WordNet ein F_1-Wert von 65.4 erzielt. Auf Wikidata wird für Wahl des ersten von der Wikidata-Suche zurückgegebenen Konzeptes ein F_1-Wert von 52.7 erzielt. Damit liegt die MFS-Baseline für den Evaluationsdatensatz nur knapp unter der für die Senseval und SemEval Datensätze. Bei der Verwendung des reinen Lesk-Algorithmus wird für WordNet ein F_1-Wert von 32.4 und für Wikidata ein F_1-Wert von 26.6 erzielt. Nimmt man das erste zurückgegebene Konzept als gesetzt an und ersetzt es wenn ein anderes eine höhere Schnittmenge der Kontexte im Sinne von Lesk hat, werden für WordNet ein F_1-Wert von 48.4 und für Wikidata ein F_1-Wert von 45.5 erzielt. Grund für die niedrigen F_1-Werte für den Lesk-Ansatz sind kleine bis gar keine Schnittmengen der Kontexte von Entität und Konzept. Dies ist der Fall, obwohl für den Kontext des Konzeptes aus der Wissensquelle alle Begriffe aus Namen und Beschreibungen von Konzepten mit der Distanz zwei in der jeweiligen Wissensquelle gewählt werden. Es scheint also sinnvoll hier andere Verfahren zu untersuchen und den Kontext auf Seiten der Entität, der momentan nur aus einem Anforderungssatz besteht, zu vergrößern.

7.4 Anwendung

Um den Mehrwert der aufgebauten Konzepthierarchien hinsichtlich der Nützlichkeit für bestimmte Aufgaben, wie der Bestimmung semantischer Beziehungen zwischen Konzepten, zu bestimmen, sind weitere aufwendige Experimente in dieser Richtung nötig. Für die Bestimmung semantischer Distanzen müsste zunächst untersucht werden welche Maße sich auf den jeweiligen Wissensquellen eignen um gute Werte für die semantische Distanz zu bestimmen. Zur letztendlichen Messung müsste ein weiterer umfangreicher Datensatz erstellt werden, der entsprechende Beziehungen für Begriffe aus Anforderungsdokumenten enthält.

8 Zusammenfassung und Ausblick

Ziel dieser Arbeit war es zu untersuchen, wie durch die Kombination mehrerer Wissensquellen ein Mehrwert bei der semantischen Untersuchungen von Begriffen aus Anforderungen erzielt werden kann. Dazu sollte eine konsolidierte Konzepthierarchie aufgebaut werden anhand derer später verschiedene Untersuchungen wie die Bestimmung von semantischen Beziehungen zwischen Konzepten und deren Zugehörigkeit zu Themengebieten untersucht werden können.

Zunächst wurden die beiden Wissensquellen WordNet und Wikidata näher betrachtet. Daraufhin wurde untersucht welche Schritte nötig sind um aus diesen und beliebigen weiteren Wissensquellen eine konsolidierte Konzepthierarchie zu erzeugen. Dafür wurde zunächst ein Modell einer Konzepthierarchie entwickelt in das beliebige Wissensquellen übersetzt werden können. Außerdem wurde für WordNet und Wikidata beispielhaft die Abbildung auf dieses Modell bestimmt. Als nächster Schritt wurde die Bestimmung der passenden Konzepte für die in Anforderungen vorkommenden Entitäten bestimmt. Da es sich bei diesen Entitäten häufig um zusammengesetzte Ausdrücke beziehungsweise Nominalphrasen gehandelt hat, war die Bestimmung der passenden Konzepte nicht unmittelbar möglich. Stattdessen wurden Regeln zur Bestimmung des lexikalischen Kopfes einer Nominalphrase dazu verwendet um ein Verfahren zur entwickeln, bei dem für die einzelnen Schritte bei der Reduktion auf den lexikalischen Kopf einer Entität ein passendes Konzept in den Wissensquellen gesucht wurde. Die Varianten der Phrase bis zur ausreichenden Reduktion zum finden eines Einstiegskonzeptes wurden dann als abgeleitete Konzepte betrachtet. Diese verbinden die in einer Anforderung vorliegende Entität mit vollständig zusammengesetztem Namen über Hyperonym-Beziehungen mit den abgeleiteten Konzepten für die einzelnen Reduktionsschritte bis zum ersten gefundenen Konzept aus einer Wissensquelle. Außerdem wurde der Fall betrachtet bei dem nur einzelne Wissensquellen ein passendes Konzept für eine Entität beinhalten. Hier wurde ein Ansatz vorgeschlagen bei dem entlang des Hyperonym-Pfades eines gefundenen Konzeptes in einer Wissensquelle A für eine Wissensquelle B ein identisches Konzept gesucht wird. Hierdurch entsteht ein Ansatz, bei dem verschiedene Wissensquellen bereits an den entsprechenden Einstiegskonzepten, also gewissermaßen an den Rändern, jeweils für jede Entität einen Einstiegspfad bieten. Dadurch lassen sich für Paare von Konzepten Beziehungen betrachten, auch wenn nicht beide in der gleichen Wissensquelle vorhanden sind. Auf Basis der Suche nach den passenden Konzepten wurde dann betrachtet, wie für jede Wissensquelle jeweils eine Konzepthierarchie aufgebaut werden kann, die die gefundenen Konzepte enthält. Dabei wurden die zwei genannten Anwendungsfälle der Untersuchung semantischer Beziehungen und das Bestim-

men von Relevanten Themen näher betrachtet und untersucht welche Konzepte aus den Wissensquellen in eine solche Konzepthierarchie übernommen werden sollten. Schließlich wurde auf Basis der einzelnen aufgebauten Konzepthierarchien untersucht wie diese zu einer Konzepthierarchie konsolidiert werden können. Dabei wurden die Techniken des Abbildens und des Vereinigens von Ontologien näher betrachtet. Dabei wurde festgestellt, dass die bisher in diesem Bereich existierenden Verfahren sich vermutlich nicht zur automatischen Vereinigung von Wissen aus allgemeinen Wissensquellen eignen. Die Betrachtung von aufgebauten Konzepthierarchien hat gezeigt, dass diese sich gerade in den höheren Ebenen der Hierarchie stark unterscheiden. Dies kommt daher, dass es hier oft kein richtig oder falsch bei der Modellierung gibt, wodurch eine Vielzahl unterschiedlicher Möglichkeiten zur Modellierung existieren. Die Vermutung liegt daher nahe, dass sich die Wissensquellen eher für heuristische Ansätze als zur vereinigten expliziten Modellierung von Wissen eigenen könnten.

An verschiedenen Stellen des vorgestellten Ablaufs zum Aufbau und zur Konsolidierung von Konzepthierarchien sind grundlegende Verbesserungen denkbar. Ein Ansatzpunkt ist dabei die flüssige Integration von ganzen Anforderungsdokumenten statt einzelnen natürlichsprachlichen Aussagen. Die Anforderungsdokumente enthalten oftmals Glossare, die gerade die komplizierteren oder abgekürzten Begriffe, bei denen die Auswahl der passenden Konzepte in den Wissensquellen schwer fällt, beschreiben. Außerdem hat sich bei der Evaluation gezeigt, dass der Kontext einer einzelnen Anforderung, zumindest für die nicht auf trainierten Modellen basierende Disambiguierung, vermutlich nicht ausreichend ist. Allerdings werden Begriffe in den jeweiligen Anforderungen oft mehrfach und möglicherweise häufig mit der gleichen Wortbedeutung wiederholt. Durch die gleichzeitige Betrachtung aller Stellen in denen ein Wort in einer bestimmten Wortbedeutung auftritt ließe sich also vermutlich der Kontext für einen Begriff wesentlich erhöhen.

Literaturverzeichnis

[ACC+02] Antoniol, G. ; Canfora, G. ; Casazza, G. ; De Lucia, A. ; Merlo, E.: Recovering traceability links between code and documentation. In: *IEEE Transactions on Software Engineering* 28 (2002), Oktober, Nr. 10, 970–983. http://dx.doi.org/10.1109/TSE.2002.1041053. – DOI 10.1109/TSE.2002.1041053. – ISSN 0098–5589 (zitiert auf Seite 1).

[BGL02] Bussmann, Hadumod [. (Hrsg.) ; Gerstner-Link, Claudia (Hrsg.): *Lexikon der Sprachwissenschaft*. 3., aktualisierte und erw. Aufl. Stuttgart : Kröner, 2002 http://www.gbv.de/dms/hbz/toc/ht013286143.pdf;http://d-nb.info/965554066/04;http://swbplus.bsz-bw.de/bsz098644408rez.htm;http://digitale-objekte.hbz-nrw.de/webclient/DeliveryManager?pid=1810670&custom_att_2=simple_viewer. – ISBN 3–520–45203–0 978–3–520–45203–0 (zitiert auf den Seiten 3, 4, 5 und 6).

[DMDH04] Doan, AnHai ; Madhavan, Jayant ; Domingos, Pedro ; Halevy, Alon: Ontology Matching: A Machine Learning Approach. Version: 2004. http://dx.doi.org/10.1007/978-3-540-24750-0. In: Staab, Steffen (Hrsg.) ; Studer, Rudi (Hrsg.): *Handbook on Ontologies*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2004. – DOI 10.1007/978–3–540–24750–0. – ISBN 978–3–662–11957–0 978–3–540–24750–0, 385–403 (zitiert auf Seite 36).

[Dvo09] Dvorak, Daniel: NASA Study on Flight Software Complexity. In: *AIAA Infotech@Aerospace Conference*. Seattle, Washington : American Institute of Aeronautics and Astronautics, April 2009. – ISBN 978–1–60086–979–2 (zitiert auf Seite 1).

[DWS17] Dorsch, Friedrich (Hrsg.) ; Wirtz, Markus A. (Hrsg.) ; Strohmer, Janina (Hrsg.): *Dorsch - Lexikon der Psychologie*. 18., überarbeitete Auflage. Bern : Hogrefe, 2017. – ISBN 978–3–456–85643–8. – OCLC: 958269363 (zitiert auf Seite 3).

[EaEE98] Electrical and Electronics Engineers, Institute of (Hrsg.): *IEEE recommended practice for software requirements specifications: approved 25 June 1998*. New York, NY : IEEE, 1998 (IEEE Std 830-1998). – ISBN 978–0–7381–0332–7. – OCLC: 254961688 (zitiert auf Seite 29).

[Fel99] Fellbaum, Christiane (Hrsg.): *WordNet: an electronic lexical database*. 2. printing. Cambridge, Mass. : MIT Press, 1999 (Language, speech, and communication). – ISBN 978–0–262–06197–1. – OCLC: 247366900 (zitiert auf den Seiten 1, 14 und 21).

[FSG17] Ferrari, Alessio ; Spagnolo, Giorgio O. ; Gnesi, Stefania: PURE: A Dataset of Public Requirements Documents. In: *2017 IEEE 25th International Requirements Engineering Conference (RE)*. Lisbon, Portugal : IEEE, September 2017. – ISBN 978–1–5386–3191–1, 502–505 (zitiert auf Seite 45).

[HYS10] Hayashi, Shinpei ; Yoshikawa, Takashi ; Saeki, Motoshi: Sentence-to-Code Traceability Recovery with Domain Ontologies. In: *2010 Asia Pacific Software Engineering Conference*. Sydney, Australia : IEEE, November 2010. – ISBN 978–1–4244–8831–5, 385–394 (zitiert auf Seite 11).

[IEE] IEEE: IEEE Guide for CASE Tool Interconnections - Classification and Description. http://dx.doi.org/10.1109/IEEESTD.2003.94231. – Forschungsbericht (zitiert auf Seite 8).

[ISO] IEEE: ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes – Requirements engineering. http://dx.doi.org/10.1109/IEEESTD.2018.8559686. – Forschungsbericht (zitiert auf den Seiten 5 und 6).

[JM09] Jurafsky, Dan ; Martin, James H.: *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition*. 2. ed., Pearson internat. ed. Upper Saddle River, NJ : Prentice Hall, Pearson Education Internat, 2009 (Prentice Hall series in artificial intelligence). – ISBN 978–0–13–504196–3. – OCLC: 263455133 (zitiert auf den Seiten 7, 22, 26 und 31).

[Kle01] Klein, Michel: Combining and Relating Ontologies: An Analysis of Problems and Solutions. (2001) (zitiert auf den Seiten xi, 8, 15, 16 und 35).

[KSY+10] Kaiya, Haruhiko ; Shimizu, Yuutarou ; Yasui, Hirotaka ; Kaijiri, Kenji ; Saeki, Motoshi: Enhancing Domain Knowledge for Requirements Elicitation with Web Mining. In: *2010 Asia Pacific Software Engineering Conference*. Sydney, Australia : IEEE, November 2010. – ISBN 978–1–4244–8831–5, 3–12 (zitiert auf Seite 17).

[LCH13] Li, Yonghua ; Cleland-Huang, Jane: Ontology-based trace retrieval. In: *2013 7th International Workshop on Traceability in Emerging Forms of Software Engineering (TEFSE)*. San Francisco, CA, USA : IEEE, Mai 2013. – ISBN 978–1–4799–0495–2, 30–36 (zitiert auf Seite 11).

[MCG16] McTear, Michael ; Callejas, Zoraida ; Griol, David: *The Conversational Interface*. Cham : Springer International Publishing, 2016. http://dx.doi.org/10.1007/978-3-319-32967-3. http://dx.doi.org/10.1007/978-3-319-32967-3. – ISBN 978–3–319–32965–9 978–3–319–32967–3 (zitiert auf Seite 6).

[NM00] Noy, Natalya F. ; Musen, Mark A.: PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. In: *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, AAAI Press, 2000. – ISBN 0–262–51112–6, 450–455. – tex.acmid: 721118 tex.numpages: 6 (zitiert auf Seite 17).

[NM03] Noy, Natalya F. ; Musen, Mark A.: The PROMPT suite: interactive tools for ontology merging and mapping. In: *International Journal of Human-Computer Studies* 59 (2003), Dezember, Nr. 6, 983–1024. http://dx.doi.org/10.1016/j.ijhcs.2003.08.002. – DOI 10.1016/j.ijhcs.2003.08.002. – ISSN 10715819 (zitiert auf den Seiten xi, 35 und 36).

[NP12] Navigli, Roberto ; Ponzetto, Simone P.: Multilingual WSD with Just a Few Lines of Code: The BabelNet API. In: *Proceedings of the ACL 2012 System Demonstrations*, Association for Computational Linguistics, 2012 (ACL '12),

67–72. – tex.acmid: 2390482 tex.numpages: 6 event-place: Jeju Island, Korea (zitiert auf den Seiten 14 und 27).

[RCCN17] Raganato, Alessandro ; Camacho-Collados, Jose ; Navigli, Roberto: Word Sense Disambiguation: A Unified Evaluation Framework and Empirical Comparison. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain : Association for Computational Linguistics, 2017, 99–110 (zitiert auf Seite 48).

[Roc03] Roche, Christophe: Ontology: A Survey. In: *IFAC Proceedings Volumes* 36 (2003), September, Nr. 22, 187–192. `http://dx.doi.org/10.1016/S1474-6670(17)37715-7`. – DOI 10.1016/S1474–6670(17)37715–7. – ISSN 14746670 (zitiert auf Seite 8).

[SHR07] Stoica, Emilia ; Hearst, Marti ; Richardson, Megan: Automating creation of hierarchical faceted metadata structures. In: *Human language technologies 2007: the conference of the north American chapter of the association for computational linguistics; proceedings of the main conference*. Rochester, New York : Association for Computational Linguistics, April 2007, 244–251 (zitiert auf Seite 32).

[Udo06] Udo Hebisch: *Ontologien*. `http://www.mathe.tu-freiberg.de/~hebisch/onto.pdf`. Version: November 2006 (zitiert auf Seite 7).

[VK14] Vrandečić, Denny ; Krötzsch, Markus: Wikidata: a free collaborative knowledgebase. In: *Communications of the ACM* 57 (2014), September, Nr. 10, 78–85. `http://dx.doi.org/10.1145/2629489`. – DOI 10.1145/2629489. – ISSN 00010782 (zitiert auf den Seiten 1, 14 und 22).

[WT15] Weigelt, Sebastian ; Tichy, Walter F.: Poster: ProNat: An Agent-Based System Design for Programming in Spoken Natural Language. In: *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*. Florence, Italy : IEEE, Mai 2015. – ISBN 978–1–4799–1934–5, 819–820 (zitiert auf Seite 9).

Anhang

A Evaluationsdatensatz - Anforderungen

ID	Anforderung
1	The actor gets one or more screens displaying one or more of the search results. For each result, the title and the main author of the publication are displayed. Navigation through different screens of search results is possible with the next or previous page option.
2	The actor can download a selected publication by clicking on the Download button for the selected publication.
3	In the detailed view screen of a publication the actor displays the full text of the publication by clicking the full text button just under the abstract.
4	The actor selects the document to add to the system and confirms. The system tries to retrieve the publication details from the document. The system shows the new publication with the retrieved information. The actor either accepts or makes some changes to the publication and confirms the changes.
5	Administrator selects the Delete Group function. Administrator fills in groupname. System shows group details. Administrator confirms.
6	The system must track the workflow between senders and related replies.
7	The system must provide the management of user rights and permissions.
8	The system must allow a member of the network to send a bulk request on all or a large part of its driver's smart card holders to a particular or all members of the network.

9	The system must allow a Member State (through its Card Issuing Authority) to send card status modification requests (lost, stolen,...) to the corresponding Card Issuing Authority of the Member State having issued the card.
10	The system must allow a member to do statistics on messages issued and received from/to his system.
11	The user shall be able to view the data stored in the Physical Inventory tables through the Pontis graphical user interface (GUI).
12	The user shall be able to find projects by project ID, project name, project status, structure ID, and program.
13	The system shall provide the ability to save reports in PDF, HTML, and XML formats.
14	The user shall be able to create new structure(s) by importing from a supported file type. The supported file types are to be determined, but will include at least PDI, NBI, and XML.
15	The system shall be capable of initiating a session without a login prompt. This capability will provide support for an independent launch shell for Pontis and will be configured by the system administrator.
16	The scheduler shall execute cyclic tasks, providing an API call to allow the application to block until its next iteration.
17	The FTSS software shall provide an API call that provides the application program the minor frame number.
18	For software exceptions occurring during Startup, FTSS shall issue a VME reset to the FCR in which the exception occurred.
19	If there is insufficient space to enqueue a message for transmission, Communication services shall return an error to the corresponding task. Sockets are non-blocking and place the burden of polling on the application task.
20	The application software shall have the capability to reset a permanently failed channel to its initial recovery state.
21	It is the first thing a user must do to begin using KeePass. Its main function is the determination of the master password that will unlock the database from now on.
22	When choosing to open a database a user is transferred to his documents where he navigates to find the database he wants. When the database is found, the master password is wanted so that the database will be unlocked. Once this is done the user is free to access his data.

23	Folder selected must be of type the database can read and that is “name”.kdb
24	When a database is opened, the user can access his passwords, organize them into new groups and subgroups, delete and add entries and so much more. But when it is time for the database to close or during his working on the database, he can save the changes made.
25	While working on the database, the user has the option to print data from his database. This can be done by selecting print. When this happens, a list of data types that can be printed are shown and the user can select the data to be printed. More specifically fields that can be selected for printing are: Backup entries, which contain entries in the back up group, password groups, group tree, title, username, password, URL, notes, creation time, last access, last modification, expires, icon, UUID and attachment.
26	The distributor will submit the sales details back to the VMU.
27	If the patient is not cured then the doctor will refer the patient to some other Hospitals the hospital may be another VSP or any other.
28	Based on the payment terms agreed by VSP, the field office will generate BiMonth or Monthly financial and medical report and send it to MSIU Admin team to arrange the payments for the VSP.
29	The voucher numbers are system generated and created with unique identification numbers with security protocols inbuilt.
30	Each voucher should contain the bar code of the Voucher with two identifications one for the client and the other for the partner.
31	If the number of block movements of the current player is lower than the highest number of block movement recorded in the statistic file, the The ”Finish Window with Statistcsïs displayed, see chapter 3.2.2.2. If not the SSimple finish Windowïs displayed, see chapter 3.2.2.3.
32	If the file was correctly updated, there is no ouput. If not, like wrong pemissions or disk full, an error message is displayed.
33	A dialog box is open : the user could choose a file that will contain all the data of the current game. Next, the following internal data are saved into the file : the current positions of the blocks, their previous positions, the number of the previous movements and the time passed by the user to solve the headache.

B Evaluationsdatensatz - Entitäten

Entität	WordNet ID	Wikidata ID	Reduziert	WordNet ID	Wikidata ID
actor	9786620				
screen					
search result		Q2686427	result	7307418	
result	7307418	Q2995644			
title	6354890	Q783521			
main author			author	10813654	Q482980
publication	6601855	Q732577			
navigation					
next page option			option	5798949	
previous page option			option	5798949	
selected publication			publication	6601855	Q732577
download button			button		Q1335171
detailed view screen			screen		
full text button			button		Q1335171
abstract	6480622	Q333291			
document	6481744	Q49848			
system	4384144	Q58778			
publication detail			detail	5825971	Q1201081
new publication			publication	5825971	Q1201081
retrieved information			information	6646883	Q11028
some change			change	7311046	Q1150070
change	7311046	Q1150070			
administrator		Q16532929			
delete group function			function	5156572	
groupname					
group detail			detail	5825971	Q1201081
workflow	251010	Q627335			
sender	10598214				
related reply	6758700	Q1920566	reply		
management	1135602				
user right			right	5182180	Q780687
permission	6702042				
system	4384144	Q58778			
member	13832827	Q9200127			
network	8451269				
bulk request			request	7199985	Q57268173
large part			part	13831419	Q15989253
smart card holder			holder	10199809	
particular member			member	13832827	Q9200127
all member			member	13832827	Q9200127
system	4384144	Q58778			
member state		Q185441	state	8671935	

Card Issuing Authority			authority	5203850	Q174834
card status modification request			request	7199985	Q57268173
card	6489042	Q42965339			
system	4384144	Q58778			
member	13832827	Q9200127			
statistic	6030848	Q1949963			
message	6263820	Q62852			
user	10761247				
data	8479331	Q42848			
inventory table	8283156	Q496946	table		
graphical user interface	6587857	Q782543			
user	10761247				
project	797381	Q170584			
project id			id		Q853614
project name			name	6344646	Q82799
project status			status		Q1796826
structure id			id		Q853614
program	5907175				
system	4384144	Q58778			
report	7233130	Q10870555			
pdf format		Q42332			
html format			format	6649331	Q235557
xml format				6649331	Q235557
user	10761247				
structure	4939142	Q6671777			
supported file type			type		Q235557
pdi		Q7231341			
nbi					
xml		Q2115			
system	4384144	Q58778			
session		Q932410			
login prompt			prompt	6291257	Q25416463
support	1218392				
independent launch shell			shell		Q14663
system administrator	10707012	Q327353			
capability	5209765	Q1347367			
scheduler	4152482	Q1123036			
cyclic task		Q759676	task	797381	
API call			call		
application	6582286	Q166142			
next iteration			iteration	13525376	Q651022
FTTS software			software	6578068	Q7397
API call			call		
application program	6582286		program		

minor frame number			number	5128718	Q11563
software exception			exception	5715216	Q779608
startup	243298				
exception	5715216	Q779608			
insufficient space			space		
message	6263820	Q133492			
transmission	122186	Q118093			
communication service			service	319217	
error	71785	Q29485			
corresponding task			task	797381	Q759676
socket		Q632343			
application task			task	797381	Q759676
application software		Q166142		6578068	
capability	5209765	Q1347367			
permanently failed channel			channel	6270551	
initial recovery state			state	24900	
user	10761247	Q278368			
main function			function	5156572	Q1724915
master password			password	6686933	Q161157
database	6650349	Q8513			
database	6650349	Q8513			
user	10761247	Q278368			
document	6522968	Q49848			
master password			password	6686933	Q161157
data	8479331	Q42848			
folder		Q201456			
type	5848697	Q190087			
database	6650349	Q8513			
database	6650349	Q8513			
user	10761247	Q278368			
password	6686933	Q161157			
group	31563	Q16887380			
subgroup	8017525				
entry	6515715				
change		Q1150070			
database	6650349	Q8513			
user	10761247	Q278368			
option to print			option	5798949	Q28827890
data	8479331	Q42848			
list of data types			list	6492991	Q12139612
data type		Q190087			
field					
backup entry			entry	6515715	
entry	6515715				
backup group			group	31563	Q16887380

password group			group	31563	Q16887380
group tree			tree	13935275	Q272735
title	6357831	Q783521			
username		Q26403005			
password	6686933	Q161157			
url	6370154	Q42253			
note	6516453				
creation time			time	15295388	Q11471
last access			access	281976	
last modification			modification	200556	
icon	7284621	Q138754			
uuid		Q195284			
attachment	2758249	Q352858			
distributor	10038586	Q60614978			
sales detail			detail	5825971	
patient	10425439	Q181600			
doctor	10040615	Q39631			
hospital	3545775	Q16917			
payment term			term	6783666	
field office			office	3847186	Q12823105
financial report		Q192907			
medical report	7234838	Q2782326			
admin team			team	8225481	Q327245
payment	1122769	Q1148747			
voucher number			number	5128718	Q11563
unique identification number			identification number		Q6545185
security protocol		Q1254335			
voucher	6530059	Q689044			
bar code	6366181	Q856			
identification	6898133	Q3265221			
client	10004189	Q852835			
partner	9954892	Q3366101			
number of block movements	5128718	Q11563	number		
current player			player	10459618	Q4197743
statistics file			file	6520807	Q82753
finish window			window	4596042	Q835016
chapter	6407785	Q1980247			
simple finish window			window	4596042	Q835016
file	6520807	Q82753			
output	7279593	Q778569			
wrong permission			permission	6702042	Q231043
disk	3712192	Q64817736			
error message		Q1332193			
dialog box	3191952	Q86915			
user	10761247	Q278368			
file	6520807	Q82753			
data	8479331	Q42848			
current game			game	456623	Q7889
internal data			data	8479331	Q42848

current position time	15295388	Q11471	position	5081943	Q17334923