JOHANNES KEPLER
UNIVERSITÄT LINZ
Netzwerk für Forschung, Lehre und Praxis

Rollen im Web of Data

Analyse, RDF-basiertes Rollenmodell und zugehöriger Data Browser

DIPLOMARBEIT

zur Erlangung des akademischen Grades
„Magister der Sozial- und Wirtschaftswissenschaften“
(Mag.rer.soc.oec.)
im Diplomstudium Wirtschaftsinformatik

Eingereicht am

Institut für Wirtschaftsinformatik -
Data & Knowledge Engineering

Eingereicht von

Michael Huemer

Begutachter

o. Univ.-Prof. Dr. Michael Schrefl

Mitbetreuung

Mag. Bernd Neumayr

Linz, im September 2009

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit mit dem Titel „Rollen im Web of Data“ selbständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benützt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Linz, im September 2009 _____

(Michael Huemer)

Danksagung

An dieser Stelle möchte ich mich bei all jenen bedanken, die durch ihre fachliche und persönliche Unterstützung zur Entstehung dieser Diplomarbeit beigetragen haben.

Herrn o. Univ.-Prof. Dr. Michael Schrefl und Herrn Mag. Bernd Neumayr möchte ich für die Unterstützung bei der Erstellung dieser Arbeit danken. Mag. Bernd Neumayr hat durch seine sehr engagierte Betreuung sowie durch seine fachlichen Ratschläge in diversen Diskussionen wesentlich zu dieser Arbeit beigetragen.

Mein besonderer Dank richtet sich an meine Eltern, die mir das Studium ermöglicht und mich nicht nur finanziell sondern auch moralisch immer wieder unterstützt haben.

Bedanken möchte ich mich auch bei meiner Schwester, meinem Schwager und meinen Freunden für ihre Unterstützung während des Studiums und dafür, dass sie für den nötigen Ausgleich gesorgt haben.

Mein besonderer Dank gilt auch Tina, für ihre Unterstützung, ihre motivierenden und aufbauenden Worte, sowie ihr Verständnis dafür, dass wir in den vergangenen Monaten nur so wenig Zeit miteinander verbringen konnten.

Kurzfassung

Im World Wide Web ist eine schier unüberschaubare Menge an Informationen nur in Form von Hypertext verfügbar und nur für die Verarbeitung durch Menschen geeignet. Maschinen haben Schwierigkeiten derartige Informationen zu erfassen, zu verarbeiten und zueinander in Beziehung zu setzen. Im Semantic Web, einer Erweiterung des World Wide Webs, sollen Inhalte so repräsentiert werden, dass sie nicht nur für Menschen verständlich sind, sondern auch von Maschinen zumindest soweit interpretiert werden können, dass eine (Teil-)Automatisierung verschiedener Aufgaben möglich wird. Das Web of Data ist eine konkrete Umsetzung der Idee des Semantic Webs und basiert auf HTTP, URIs und RDF. Linked Data Browser, wie Tabulator, erlauben die Erkundung und Analyse der im Web of Data verteilten und verlinkten Daten durch menschliche Benutzer.

Rollen gelten in der objekt-orientierten Modellierung als Konstrukte um Phänomene aus der Realität intuitiver abzubilden und um sich weiterentwickelnde Objekte aus der realen Welt zu verwalten. Manche Ansätze interpretieren eine Rolle als eine Spezialisierung eines Objekttyps (z.B. Employee als Subklasse von Person); andere Ansätze modellieren eine Rolle als eine Zusatzinstanz zur Objektinstanz (z.B. Employee als Zusatzinstanz zu Person). Während es Auffassungsunterschiede zu Rollen gibt, ist deren Bedeutung als wichtiges Konstrukt in der objekt-orientierten Modellierung unbestritten. Dennoch sind Rollen, meines Wissens nach, im Kontext von Web of Data und Semantic Web bisher nicht diskutiert worden.

In dieser Arbeit wird ein RDF-basiertes Rollenmodell für das Web of Data entwickelt. Dazu wird zuerst ein Rollenmodell aus der objekt-orientierten Welt analysiert, um als Vorbild für das Rollenmodell im Web of Data zu dienen. Es werden die unterschiedlichen Rahmenbedingungen im Web of Data und in der objekt-orientierten Welt gegenüber gestellt. Weiters werden Anwendungsfälle für das Rollenmodell im Web of Data beschrieben. Aus den Rahmenbedingungen und den Anwendungsfällen werden Anforderungen an das Rollenmodell abgeleitet. Nach einer abstrakten Beschreibung des Rollenmodells wird dessen konkrete Umsetzung als Rollenvokabular in RDF gezeigt. Abschließend wird beschrieben, wie der Linked Data Browser Tabulator im Rahmen dieser Arbeit um das Rollenmodell erweitert worden ist; damit wird auch gezeigt, wie um Rollen erweiterte RDF-Daten visualisiert und erkundet werden können.

Abstract

In the World Wide Web a mass of information is only available as hypertext. While perfectly suitable for human consumption, machines have difficulties to process such information. In the Semantic Web, an extension of the World Wide Web, content is represented in a way that facilitates automatic processing of data and, thus, allows (semi-)automation of various tasks. A concrete implementation of the Semantic Web is the Web of Data. It is based on HTTP, URIs and RDF.
Linked Data Browsers, such as Tabulator, allow users to explore and to analyse distributed and linked data on the Semantic Web.

In object-oriented modeling and object-oriented systems, roles are considered as an important construct for representing different aspects of real world objects as well as a means for managing evolving objects.
Some approaches represent roles as specialization of object types (e.g. employee as subclass of person); other approaches represent them as adjunct objects (e.g. employee as adjunct to person). Although there is no consensus about the details of role representation, their importance is widely acknowledged. However, to the best of my knowledge, roles have not been discussed in the context of Semantic Web and Web of Data.

In this thesis, an RDF-based role model for the Web of Data is introduced. For that purpose a role model for object-oriented systems is analysed and serves as basis for the RDF-based role model. The different basic conditions in object-oriented systems and in the Web of Data are contrasted. Furthermore, use cases for roles in the Web of Data are defined in order to, then, derive requirements for the role model. After conceptually introducing roles for the web of data, its implementation as RDF vocabulary is introduced and exemplified. Finally, an extension of the Linked Data Browser Tabulator is presented. This extended data browser is used to visualize and explore role-enhanced RDF-data.

Inhaltsverzeichnis

1. Einleitung 13
1.1 Aufgabenstellung und Zielsetzung 15
1.2 Durchgängiges Beispiel 15
1.3 Aufbau der Arbeit 16
Teil A – Grundlagen 18
2. Grundlagen Semantic Web 19
2.1 Das Semantic Web 19
2.2 Der Semantic Web Stack 20
2.3 Semantic Web vs. Web 2.0 23
2.4 Linked Data – Web of Data – Semantic Web 23
2.5 Linked Data Technologien 25
2.5.1 Ressourcen 25
2.5.1.1 Informationsressourcen 26
2.5.1.2 Nicht-Informationsressourcen 27
2.5.2 Uniform Resource Identifier (URI) 29
2.5.2.1 Zusammenhang zwischen Ressource und URI 30
2.5.2.2 Autoritative und nicht-autoritative Beschreibungen 32
2.5.3 Resource Description Framework (RDF) 32
2.5.4 Resource Description Framework Schema (RDFS) 34
2.5.4.1 Klassifizieren mit RDF(S) 35
2.5.4.2 Klassenhierarchien mit RDF(S) 36
2.5.4.3 Prädikathierarchien mit RDF(S) 37
2.6 URIs und Identität – das Coreferenz-Problem 37
2.6.1 Empfehlung der Linked Data Community 38
2.6.2 Ansatz nach [JGM07] und [JGM08] 39
2.6.3 Ansatz nach [BSCT08] 41
2.6.4 Zusammenfassung 42
3. Rollen von Objekten in der objekt-orientierten Modellierung 44
3.1 Einleitung und Überblick 44
3.1.1 Rollen als benannte Stelle 45
3.1.2 Rollen als Spezialisierung und/oder Generalisierung 45
3.1.3 Rollen als Zusatzinstanzen 46

3.2 Ausgewählte Rollenmodelle aus der objekt-orientierten Modellierung 46
3.2.1 Das Rollenmodell nach [GSR96] 47
3.2.1.1 Eigenschaften von Rollen nach [GSR96] 47
3.2.1.2 Rollenhierarchien im Ansatz von [GSR96] 48
3.2.1.3 Qualifizierte Rollen 49
3.2.1.4 Kombination von Klassen- und Rollenhierarchien 50
3.2.2 Das Rollenmodell nach [WJS95] 51
Teil B – Rollen im Web of Data 53
4. Rollen im Web of Data – Motivation, Analyse, Anforderungen 54
4.1 Motivation für Rollen im Web of Data 54
4.1.1 Argument a): Wesentliche Erweiterung der Ausdrucksstärke 54
4.1.2 Argument b): Beitrag zur Lösung des Coreferenz-Problems 58
4.1.3 Argument c): Förderung der Navigierbarkeit 58
4.2 Anforderungen an Rollen im Web of Data 59
4.2.1 Semantische Anforderungen 59
4.2.2 Situationsbedingte Anforderungen 59
4.2.2.1 Neuerstellung einer Rollenbeschreibung 59
4.2.2.2 Erstellung einer Spezialisierungsrolle 60
4.2.2.3 Erstellung einer Generalisierungsrolle 62
4.2.2.4 Anreicherung von vorhandenen Ressourcen 65
4.2.3 Zusammenfassung der Anforderungsdefinitionen 66
4.3 Analyse der Rahmenbedingungen im Web of Data vs. objekt-orientierter Welt 68
4.4 Entwicklung eines Rollenmodells für das Web of Data 71
4.4.1 Zwei Identitätslevel: Entitätslevel und Rollenlevel 72
4.4.2 Allgemeines zu Rollen im Web of Data 74
4.4.3 Simple Rollen 74
4.4.4 Qualifizierte Rollen 76
4.4.5 Simple und qualifizierte Rollen mit Generalisierungsinformationen 77
4.4.6 Restriktionen für Rollen 80
4.5 Das Rollenmodell – Beurteilung und Vergleich 81
4.5.1 Rollen im Web of Data vs. Rollen in [GSR96] 81
4.5.2 Navigierbarkeit zwischen Rollen 81
4.5.3 Coreferenz-Problem 83

4.5.4 Vordefinierte Rollenklassenhierarchien 84
5. Rollen im Web of Data – Umsetzung mit RDF(S) 85
5.1 Rollenvokabular 85
5.1.1 Rollenvokabular für simple und qualifizierte Rollen 85
5.1.2 Rollenvokabular für Generalisierungsinformationen auf Instanzebene 87
5.1.3 Rollenvokabular für Generalisierungsinformationen auf Schemaebene 89
5.2 Umsetzung der Anwendungsfälle im Web of Data 91
5.2.1 Erstellung einer Rollenklassenhierarchie 92
5.2.2 Anwendungsfall 1: Neuerstellung einer Rollenbeschreibung 93
5.2.3 Anwendungsfall 2: Rollen als Spezialisierung 94
5.2.4 Anwendungsfall 3: Rollen als Generalisierung 95
5.2.5 Anwendungsfall 4: Anreicherung von vorhandenen Ressourcen 98
5.3 Maßnahmen zur Verbesserung der Navigierbarkeit 99
Teil C – Visualisierung und Browsing von Rollen mittels Tabulator 101
6. Der Linked Data Browser Tabulator 102
6.1 Einleitung 102
6.2 Architektur von Tabulator 103
6.2.1 Asynchronous Javascript and RDF (AJAR) 103
6.2.2 Interner RDF-Quadruple-Store 104
6.3 Benutzerschnittstelle 105
6.3.1 Erkundungsmodus 106
6.3.2 Analysemodus 107
6.4 Ablauf einer Darstellung 110
7. Der rollenfähige Tabulator 112
7.1 Adaptierung von Tabulator 112
7.2 Rollenfähige Benutzerschnittstelle 115
8. Zusammenfassung 118
9. Literaturverzeichnis 120
10. Anhang 127
A.1 Rollenvokabular 127
A.2 Rollenhierachie 129
A.3 Lösung Anwendungsfall 1: Neuerstellung einer Rollenbeschreibung 130
A.4 Lösung Anwendungsfall 2: Rollen als Spezialisierung 131
A.5 Lösung Anwendungsfall 3: Rollen als Generalisierung 133

A.6 Lösung Anwendungsfall 4: Anreicherung von vorhandenen Ressourcen 135
A.7 Dokumentation der Änderungen im Sourcecode von Tabulator 137
A.8 Für Entwickler: Eine kurze Einführung in den Sourcecode von Tabulator 139

Abkürzungen

AI	Artificial Intelligence
AJAR	Asynchronous Javascript and RDF
AJAX	Asynchronous Javascript and XML
bzw.	beziehungsweise
CRS	Consistent Reference Service
DNS	Domain Name System
d.h.	das heißt
ENS	Entity Name System
FOAF	Friend of a Friend
HTTP	Hypertext Transfer Protocol
IRI	Internationalized Resource Identifier
JSP	Java Server Pages
lt.	laut
NIR	Nicht-Informationsressource
OWL	Web Ontology Language
PHP	PHP: Hypertext Preprocessor
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RIF	Rule Interchange Format
SPARQL	SPARQL Protocol and RDF Query Language
UML	Unified Modelling Language
UNA	Unique Name Assumption
URI	Unified Resource Identifier
vgl.	vergleiche
W3C	World Wide Web Consortium
WWW	World Wide Web
XML	Extensible Markup Language
XSL	Extensible Stylesheed Language
z.B.	zum Beispiel
vs.	versus

Abbildungsverzeichnis

Abbildung 1: Semantic Web Stack [Ber07b] 22
Abbildung 2: Content Negotiation am Beispiel einer URI welche eine Informationsressource identifiziert [BCH07] [HTTP99] 27
Abbildung 3: Anfrage einer 303 URI [BCH07] [HTTP99] 28
Abbildung 4: Zusammenhang zwischen NIR und URI lt. [BHL01] 30
Abbildung 5: Zusammenhang zwischen URI und Nicht-Informationsressource (NIR) 31
Abbildung 6: RDF-Tripple 34
Abbildung 7: Verlinkung von URIs mit owl:sameAs nach [BHL01] 38
Abbildung 8: Verlinkung von URIs mit owl:sameAs angepasst an Aussagen von [Boo06] 39
Abbildung 9: Bündelung mehrerer URIs mittels Consistent Reference Service 40
Abbildung 10: Ebenen der Identitätsprüfung 43
Abbildung 11: Rollenhierarchie im Rollenansatz von [GSR96] 49
Abbildung 12: Kombination von Rollen- und Klassenhierarchie im Ansatz von [GSR96] 51
Abbildung 13: Rollenmodell im Web of Data 56
Abbildung 14: Rollen als n-äre Beziehung am Beispiel von Mr. Black in den Rollen als Employee 57
Abbildung 15: Neuerstellung einer Rollenbeschreibung 60
Abbildung 16: Erstellung einer Spezialisierungsrolle 62
Abbildung 17: Neuerstellung einer Generalisierungsrolle 63
Abbildung 18: Neuerstellung einer Generalisierungsrolle 64
Abbildung 19: Anreicherung von vorhandenen Ressourcen 66
Abbildung 20: Entitätslevel und Rollenlevel 73
Abbildung 21: Simple Rollen 76
Abbildung 22: Qualifizierte Rollen 77
Abbildung 23: Simple Role mit Aufwärtsvererbung 79
Abbildung 24: Möglichkeiten der Aufteilung einer Rollenbeziehung (vgl. Abbildungen 15-19) 82
Abbildung 25: Übersicht der Begrifflichkeiten und des Vokabulars der Generalisierungsvererbung 91
Abbildung 26: Repräsentation einer Rollenklassenhierarchie mit rrdf Vokabular (links) und als Baum (rechts) 92

Abbildung 27: Anwendungsfall 1 – Neuerstellung von Rolleninstanzen (links) und hierarchische Darstellung dieser als Baum (rechts) 93
Abbildung 28: Anwendungsfall 2 – Rolleninstanzen als Spezialisierung in N3 Syntax (links) und als Baum (rechts) 95
Abbildung 29: Anwendungsfall 3 – Rolleninstanzen als Generalisierung in N3 Syntax (links) und als Baum (rechts) 97
Abbildung 30: Anwendungsfall 4 – Anreicherung von Rolleninstanzen in N3 Syntax (links) und als Baum (rechts) 99
Abbildung 31: Sequenzdiagramm AJAR 104
Abbildung 32: Datenstruktur in Tabulator 105
Abbildung 33: Tabulator: Erkundungsmodus 107
Abbildung 34: Analysemodus: SPARQL Abfrage 108
Abbildung 35: Analysemodus: Table-View 109
Abbildung 36: Analysemodus: Map-View 110
Abbildung 37: Aktivitätsdiagramm für den Anwendungsfall „Eingabe einer URI von einer NIR“ 111
Abbildung 38: Aktivitätsdiagramm: Verfeinerung der Aktivität „Darstellung Vervollständigen“ 114
Abbildung 39: Role-ready Tabulator: Bsp. Mr. Black mit aufgeklappten Rollen 116
Abbildung 40: Role-ready Tabulator: Bsp. Mr. Black mit zugeklappten Rollen 117

Tabellenverzeichnis

Tabelle 1: Rahmenbedingungen in [GSR96] vs. Rahmenbedingungen im Web of Data 69

1. Einleitung

Das WWW (World Wide Web) oder kurz Web genannt, ist entwickelt worden um als Informationsplattform zu dienen, an der Mensch und Maschine gleichermaßen teilhaben können [Ber98]. Allerdings hat sich das Web sehr stark nur in eine Richtung entwickelt, nämlich dahingehend, dass die Präsentation der Informationen bestmöglich vom Menschen konsumiert und genutzt werden kann. So haben sich aufgrund der Organisation und der zugrunde liegenden Infrastruktur des Webs, neben den vielen Vorteilen dessen gegenüber herkömmlichen Methoden des Informationsaustausches, wie beispielsweise die Möglichkeit der ständigen Aktualität und universellen Verfügbarkeit von Informationen, auch eine Reihe an Problemen ergeben. [BHL01], [HKRS08], [Zam05], [SHB06], [AnHa08] und [ToMa06] haben die Ursachen und Folgen der Probleme analysiert, die ausschlaggebend für die Weiterentwicklung vom Web (1.0) zum Semantic Web sind.

Dadurch, dass eine schier unüberschaubare Menge an Informationen für eine Darstellung für den Menschen bestimmt ist, existieren diese oft nur in HTML-Dokumenten eingebettet. Für den menschlichen Nutzer ist die Darstellung der HTML-Dokumente im Webbrowser optimal und man kann die Bedeutung der dargestellten Informationen problemlos erfassen, zu anderen Informationen in Relation setzen oder falls notwendig in eine andere Darstellungsform transformieren. Für Maschinen sind diese Vorgänge schon erheblich schwieriger, falls überhaupt möglich. Daraus ergibt sich das Problem, dass Informationen die im Web zwar grundsätzlich vorhanden und verfügbar sind in der Fülle der vorhandenen Informationen nur schwer zu finden und auszuwerten sind. Selbst für hoch leistungsfähige Suchmaschinen wie beispielsweise Google[1] oder Yahoo[2], die mit ausgefeilten statistischen Methoden arbeiten, ist es sehr schwierig alle bzw. nur relevante Informationen zu liefern, da die Suche letztendlich darauf basiert, Zeichenketten in HTML-Dokumenten zu lokalisieren. Im Ergebnis brauchbarer wäre eine kontextbezogene und weniger stichwortbasierte Suche: eine semantische Suche.
Vielfach ist auch der Zugriff auf Daten im Web aufgrund der eingesetzten Technologie nicht möglich. Beispielsweise liegen aus organisatorischen Gründen wertvolle Daten in Datenbanken versperrt, welche bei Bedarf dynamisch in Dokumente eingebettet und

[1] www.google.com
[2] www.yahoo.com

dargestellt werden. Dies erschwert es Suchmaschinen noch zusätzlich zu den relevanten Informationen zu kommen.
Durch die dezentrale Organisation und Struktur des Webs haben sich heterogene Standards auf vielen Ebenen entwickelt, beispielsweise was die Kodierungstechniken (ASCII oder Unicode), verwendete natürliche Sprachen oder verwendete serverseitige Technologien (PHP, JSP, etc.) angeht. Dies macht es, falls überhaupt möglich, sehr schwierig verteilte Informationen zu integrieren.
Weiters ist es aufgrund der fehlenden semantischen Strukturierung der Informationen im Web nicht bzw. nur sehr schwer möglich, von Informationen die zwar explizit im Web verfügbar und auffindbar sind, Implizites abzuleiten. Beispielsweise, wenn in verteilten Dokumenten verpackt die Information steckt, dass Linz die Landeshauptstadt von Oberösterreich ist und Oberösterreich in Österreich liegt, ist es ohne semantische Technologien nahezu unmöglich zu folgern, dass Linz in Österreich liegt.

Die genannten Probleme haben zur Weiterentwicklung des Webs geführt: von für den Menschen bestimmte Dokumente, hin zu Dokumenten, die mit Daten und Informationen für Maschinen angereichert sind: das Semantic Web.

Während im WWW Dokumente verlinkt werden und dieses auch als Web of Documents bezeichnet wird, werden im Semantic Web Daten verlinkt. Analog dazu hat die Linked Data Community rund um Tim Berners-Lee, der als Erfinder des WWW gilt, den Begriff Web of Data geprägt [Hea09]. Tim Berners-Lee sieht in der Umsetzung des Semantic Web nichts anderes, als ein Web of Data [Ber98]. Auf eine diesbezügliche begriffliche Abgrenzung wird später genauer eingegangen (vgl. Kapitel 2.4).

Im Web of Data werden unter anderem Objekte aus der realen Welt beschrieben. Ein Autor[3] einer solchen Beschreibung muss dabei nur sehr wenige Bedingungen einhalten (vgl. Kapitel 2.4). Diese Bedingungen machen beispielsweise weder eine Aussage darüber welche Eigenschaften des Objektes aus der realen Welt beschrieben werden müssen, noch besagen sie welches Vokabular dafür zu verwenden ist. D.h. Beschreibungen im Web of Data sind abhänigig vom Autor. Er alleine bestimmt welches Vokabular er verwendet und in welchem Kontext er etwas beschreibt. Der Kontext einer Beschreibung ist unter

[3] Da in der deutschen Sprache durch den generischen Maskulin beide Geschlechter gleichermaßen miteinbezogen werden, wird in dieser Arbeit auf ein angehängtes „Innen“ und dergleichen verzichtet.

anderem ausschlaggebend dafür, ob jemand auf dieses Modell einer Entität, wie [Boo06] es nennt, verweist oder nicht. Das Web of Data, bzw. RDF(S) als Sprache in diesem, bietet keine Möglichkeit Modelle entitätsspezifisch zu bündeln und gleichzeitig zwischen kontextspezifischen Aussagen zu unterscheiden. Es wäre hilfreich und würde die Wirklichkeit realitätsgeteuer abbilden, wenn es eine entitätsspezifische Bündelung von kontextspezifischen Modellen gäbe. Ein Rollenmodell für das Web of Data würde unter anderem dies unterstützen. In der objekt-orientierten Welt sind Rollen als intuitives und wichtiges Konstrukt zum Modellieren von Phänomenen aus der realen Welt erkannt worden [ZhZh08]. Bei der Erstellung eines Rollenmodelles für das Web of Data werden diese deshalb als Grundlage herangezogen.

1.1 Aufgabenstellung und Zielsetzung

Aufgabenstellung dieser Arbeit ist die Entwicklung eines Rollenmodells für das Web of Data. Bereits existierende Rollenmodelle aus der objekt-orientierten Welt sollen analysiert und eines davon ausgewählt werden, welches als Vorbild bei der Erstellung des Rollenmodells für das Web of Data dienen wird. Dabei soll besondere Rücksicht auf die Bedingungen im Web of Data genommen werden und bei der Erstellung von Daten nicht nur eine Top-Down Vorgehensweise unterstützt werden, sondern auch eine Bottom-Up Vorgehensweise. Das Rollenmodell soll zum konzeptionellen Modell außerdem noch ein geeignetes Rollenvokabular bieten, um Rollen mit RDF(S) abbilden zu können. Zur Darstellung von Rollen soll weiters der Linked Data Browser Tabulator dahingehend adaptiert werden, dass Rollen ihrer Semantik entsprechend dargestellt werden.

Das Ziel dieser Arbeit ist, zu zeigen, dass ein Rollenmodell einen wesentlichen Mehrwert für das Web of Data bringt. Dafür soll das entwickelte Rollenmodell beispielhaft im Web of Data umgesetzt und die sich dadurch ergebenden Vorteile analysiert werden. Der adaptierte, rollenfähige Linked Data Browser Tabulator soll bei der Darstellung die Vorteile, die man durch den Einsatz von Rollen hat, bestmöglich widerspiegeln.

1.2 Durchgängiges Beispiel

Um Rollenmodelle und andere ähnliche Ansätze zu erklären und besser zu veranschaulichen wird ein durchgängiges Beispiel verwendet. Dadurch, dass Rollenmodelle meist Phänomene aus der Wirklichkeit abbilden, wird an dieser Stelle folgendes Beispiel eingeführt:

Peter Black arbeitet bei der UniXY und verdient „1000“. In dieser Anstellung ist er als Projektmanager im Projekt „SemTech“ für die Qualitätssicherung verantwortlich. Weiters studiert er an der UniXY Informatik und hat die Matrikelnummer „0356789“. Mr. Black arbeitet auch bei der CompZ, wo er „1500“ verdient. In seiner Rolle als Privatperson ist er für den Abwasch zuständig.

Aus diesem Beispiel geht hervor, dass Mr. Black verschiedene Rollen einnimmt, wobei eine Rolle sogar doppelt eingenommen wird. Die Rolle als Student, als Privatperson und als Projektmanager wird nur einfach eingenommen, während die Rolle als Angestellter doppelt eingenommen wird, einmal bei der UniXY und einmal bei der CompZ.

1.3 Aufbau der Arbeit

Die vorliegende Arbeit ist in drei Teile gegliedert.

Teil A - Grundlagen:
In Kapitel 2 wird auf Grundlagen des Semantic Webs eingegangen. Dazu wird zuerst ein Überblick über das Semantic Web und über die Semantic Web Technologien gegeben. Nach einer begrifflichen Abgrenzung von Semantic Web, Web of Data und Linked Data wird auf ausgewählte und im Rahmen dieser Diplomarbeit relevante Technologien eingegangen. Anschließend wird auf das Identitätsproblem im Web of Data eingegangen und es werden Ansätze zur Lösung dieses Problems präsentiert.
Kapitel 3 beschäftigt sich mit Grundlagen zu Rollen und Rollenmodellen und stellt ausgewählte Rollenmodelle, welche als Basis für die weiteren Kapitel dienen, vor.

Teil B – Rollen im Web of Data:
In Kapitel 4 werden nach der Motivation für Rollen im Web of Data Anwendungsfälle für das Rollenmodell im Web of Data definiert. Aufgrund dieser Anwendungsfälle werden Anforderungen an ein Rollenmodell im Web of Data abgeleitet. Anschließend werden die Rahmenbedingungen für ein Rollenmodell aus der objekt-orientierten Welt mit denen im Web of Data gegenübergestellt und analysiert. Darauf folgend wird das Rollenmodell für das Web of Data entwickelt, anschließend beurteilt und verglichen.
In Kapitel 5 wird ein Rollenvokabular für die Umsetzung des Rollenmodells in RDF(S) präsentiert und beispielhaft auf die zuvor definierten Anwendungsfälle angewandt.

Teil C – Visualisierung und Browsing von Rollen mittels Tabulator:
In Kapitel 6 wird der Linked Data Browser Tabulator vorgestellt und beispielhaft auf wesentliche Aspekte der Architektur und Abläufe dieses eingegangen.
In Kapitel 7 wird der adaptierte rollenfähige Tabulator vorgestellt.

Kapitel 8 beinhaltet eine abschließende Betrachtung, zeigt Einschränkungen dieser Arbeit und gibt einen Ausblick.

Teil A – Grundlagen

2. Grundlagen Semantic Web 19
3. Grundlagen Rollen 44

2. Grundlagen Semantic Web

Dieses Kapitel gibt eine Einführung in die Themenbereiche Semantic Web und in ausgewählte Technologien dieses. Zuerst wird das Semantic Web als Weiterentwicklung des World Wide Web betrachtet. Nach einer begrifflichen Auseinandersetzung zwischen Semantic Web, Linked Data und Web of Data wird auf Basistechnologien eingegangen. Anschließend wird ein Exkurs zum Identitätsproblem im Semantic Web gemacht und Lösungsansätze für dieses präsentiert.

2.1 Das Semantic Web

Um der in der Einleitung beschriebenen Probleme des Webs Herr zu werden, gibt es nach [HKRS08] zwei Herangehensweisen. Die erste wäre Methoden der Künstlichen Intelligenz anzuwenden, welche die Vorgehensweise des Menschen bei der kognitiven Verarbeitung der Dokumente als Vorbild hat und die Informationen aus vorrangig für den Menschen bestimmten Dokumenten in für Maschinen verarbeitbare Daten umwandelt. Dies ist jedoch in der benötigten Qualität und im benötigten Umfang trotz einer über 50 jährigen Forschungstätigkeit im Bereich der Künstlichen Intelligenz (noch) nicht möglich.

Die zweite Herangehensweise besteht im Ansatz des Semantic Web. Tim Berners-Lee gilt als „Erfinder“ des Webs und ist Vorsitzender des W3C[4], dem Standardisierungsgremium für Web-Technologien. Er sieht in [BHL01] das Semantic Web als eine Erweiterung des Webs in welchem Informationen eine definierte Bedeutung gegeben wird, um die Mensch-Maschine Kooperation zu fördern. Später erweitert er in [SHB06] die Definition des Semantic Webs:

„The Semantic Web is a Web of actionable information – information derived from data through a semantic theory for interpreting the symbols. The semantic theory provides an account of „meaning“ in which the logical connection of terms establishes interoperability between systems.“

[ToMa06] sehen die Grundlegende Idee des Semantic Web darin, “Inhalte im Web so anzureichern, dass sie nicht nur für Menschen verständlich sind, sondern auch von Maschinen zumindest soweit erfasst werden können, dass Automatisierung auch auf Ebene der Bedeutung möglich wird”. Sie sehen im Versuch Maschinen auf semantischer Ebene arbeiten zu lassen nichts Neues, denn damit hat sich die Disziplin der Künstlichen Intelligenz (AI) schon ausgiebig beschäftigt. Auch die halb- oder vollautomatische

[4] World Wide Web Consortium

Extraktion von Semantik aus gegebenen, nicht oder nur teilweise strukturierten Datenbeständen stellt nichts Neues dar; auch damit hat sich AI schon beschäftigt. Ebenso die Organisation der Informationen, nämlich die Verteiltheit dieser ist nicht neu, denn darauf basiert schon der Vorgänger des Semantic Web, das World Wide Web. Einzig die Kombination dieser drei Elemente wird als neu gesehen.

Um die in [SHB06] geforderte „interoperability“ (Interoperabilität) zu gewährleisten, sind einheitliche und offene Standards notwendig. Unter Interoperabilität versteht man die Fähigkeit „ … Informationen zwischen verschiedenen Anwendungen und Plattformen auszutauschen und zueinander in Beziehung zu setzen … “ [HKRS08]. Das bedeutet, Informationen unabhängig von Plattform und Anwendung sollen integriert werden können. Aus diesem Anlass hat das W3C[5] 1997 die erste RDF (Resource Description Framework) Spezifikation herausgegeben. Die Vision von RDF ist eine minimale Wissensrepräsentation als Erweiterung des Webs mittels Verwendung von URIs (Unified Resource Identifiers). Somit wird jede Ressource über die URI eindeutig identifizierbar gemacht und mittels Verlinkungen dieser entsteht der Netzwerkeffekt des Webs. RDF wurde weiterentwickelt und so wurde 2004 RDFS eingeführt, welches RDF um die Möglichkeit strukturierte Vokabulare auszudrücken erweitert. Für jene, die eine mächtigere Ausdrucksmöglichkeit bei der Entwicklung ihrer Konzepte benötigen, stellt die Web Ontology Language (OWL) [DeSc04] eine Möglichkeit dar. Dass eine möglichst mächtige und ausdruckstarke Sprache nicht immer von Vorteil ist, wie auch im Semantic Web, lässt sich laut [SHB06] mit dem „principle of least power“ begründen, welches besagt: „… the less expressive the language, the more reusable the data.“ Aus gegebenem Anlass werden deshalb Daten im Linking Open Data Project (vgl. Kapitel 2.4) nur mittels RDF(S) Statements kodiert [SHB06] [Lin09].[6]

2.2 Der Semantic Web Stack

Im Semantic Web Stack (siehe Abbildung 1) werden die standardisierten Semantic Web Sprachen in Schichten angeführt. Jede Schicht baut auf den darunter liegenden Schichten auf. So wird ersichtlich, wie Technologien für das Semantic Web standardisiert und organisiert sind um das Semantic Web möglich zu machen. Des Weiteren lässt sich

[5] http://www.w3.org/
[6] Mit Ausnahme von owl:sameAs und owl:[Inverse]FunctionalProperty

erkennen, dass Semantic Web nicht das traditionelle Web ersetzt, sondern es erweitert. [AnHa08] [HPPH05]
Im Jahr 2002 wurde von Tim Berners-Lee der erste Semantic Web Stack vorgestellt. Aufgrund der voranschreitenden Forschung und Konkretisierung vieler Ebenen hat sich der Semantic Web Stack weiterentwickelt [HPPH05]. Auch die derzeitige Version des Semantic Web Stacks ist nicht als endgültig zu betrachten. Die Technologien vom Grund des Stacks bis zu OWL sind derzeit (Stand 2009) standardisiert und in Applikationen akzeptiert. Jedoch ist noch nicht ganz klar, wie der Rest der Schichten implementiert wird.

Die Basis des Semantic Web Stack bilden Unified Resource Identifier (URI) bzw. Internationalized Resource Identifier (IRI). Sie bieten die Möglichkeit Ressourcen im Semantic Web zu benennen und diese eindeutig zu identifizieren. In der zweiten Schicht befindet sich die Extensible Markup Language (XML), welche das Serialisierungsformat für die darüber liegenden Schichten, RDF und RDFS und OWL liefert. RDF und RDFS sind Sprachen um Ressourcen im Semantic Web zu verlinken. Auf URIs, RDF und RDFS, die als Basistechnologien für das Web of Data gelten, wird im Laufe dieser Arbeit noch genauer eingegangen. Die Web Ontology Language (OWL) auf der darüber liegenden Ebene erweitert RDF(S) und bietet mächtigere Konstrukte um beispielsweise Kardinalitäten oder Eigenschaften von Prädikaten wie Transitivität oder Reflexivität auszudrücken. [AnHa08] [DMH+00]

Mit der SPARQL Protocol and RDF Query Language (SPARQL) können Abfragen auf RDF basierende Daten gemacht werden. Es ist quasi die Abfragesprache für das Semantic Web. Auf SPARQL wird in dieser Arbeit nicht weiter eingegangen. Der interessierte Leser sei an [AnHa08], [Har05], [KBM08] und [PAG06] verwiesen.

Alle weiteren Schichten sind noch nicht standardisiert und enthalten zum Teil nur Ideen wie sie implementiert werden sollen.
Das Rule Interchange Format (RIF) ist ein Standard für den Austausch von verschiedenen Regelsprachen und Inferenzsystemen [BKPP07]. Auf RIF wird in dieser Arbeit nicht weiter eingegangen. Für den interessierten Leser sei an dieser Stelle sei auf [AnHa08], [BKPP07] und [KBBF05] verwiesen.

Kryptographie spielt auch im Semantic Web im Hinblick auf Security und Privacy eine wesentliche Rolle und wird vor allem im Zusammenhang mit Semantic Web Services diskutiert (vgl. [BBZF+05] und [KPSD+04]).

Das Ziel der Ebene Unifying Logic ist es, Unterschiede der darunter liegenden Logiken aufzuheben. Für eine kritische Diskussion dieses Ziels sei auf [HPPH05] und [KBBF05] verwiesen.

Die Proof-Schicht beschäftigt sich mit dem Nachweisen von Aussagen. Dies kann entweder eine URI der Informationsressource sein, der das Statement entstammt oder das Aufzeigen eines Reasoning-Vorgangs, der zum Entstehen des Statements geführt hat [BHL01].

Die Trust-Schicht beschäftigt sich mit der Vertrauensevaluierung von Ressourcen und macht Aussagen über die Vertrauenswürdigkeit von einzelnen Statements [OAKS04]. Dafür gibt es verschiedene Ansätze, beispielsweise eine inhaltsbezogene Vertrauensevaluierung [CBHS05] oder eine benutzerbezogene Vertrauensevaluierung [RAD03].

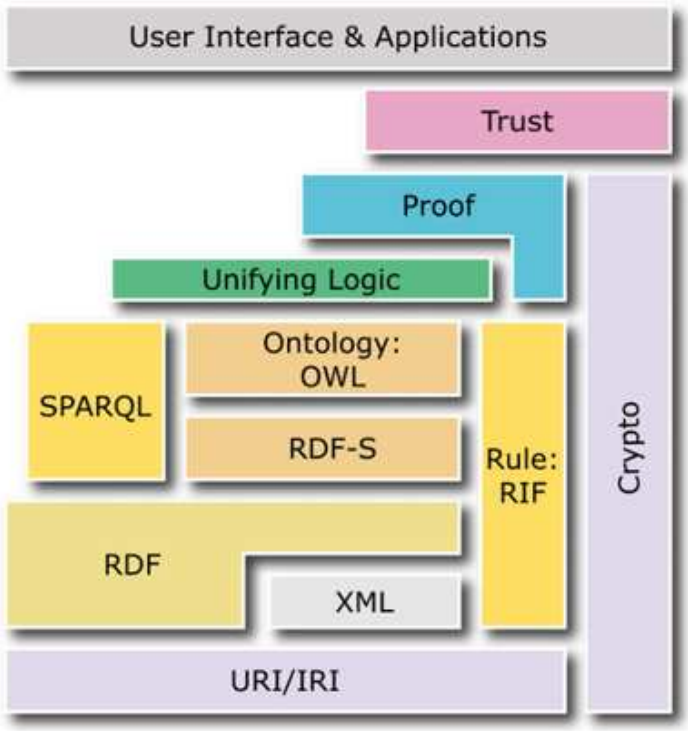


Abbildung 1: Semantic Web Stack [Ber07b]

2.3 Semantic Web vs. Web 2.0

Oftmals wird Semantic Web gleichgesetzt mit Web 2.0 (auch Social Net genannt). Ohne Zweifel besitzen die beiden Gemeinsamkeiten, wie gemeinsam genutzte Technologien und Standards (beispielsweise FOAF, XSL, XML). Jedoch unterscheiden sich das Semantic Web und Web 2.0 am primären Ziel. Während es im Web 2.0 um die Erweiterung des Webs zum Vorteil des menschlichen Benutzers geht und darum, wie diese miteinander interagieren, zusammenarbeiten, Informationen erstellen und teilen, geht es beim Semantic Web in erster Linie um die standardisierte Repräsentation von Daten, um diese für Maschinen besser nutzbar zu machen. [Zam05]
Das heißt, dass sowohl das Web 2.0, als auch das Semantic Web, auf dem Web 1.0 basieren und dieses erweitern, jedoch mit unterschiedlichem Fokus.

2.4 Linked Data – Web of Data – Semantic Web

In der Wissenschaft gibt es keine Einigkeit darüber wie die drei Begriffe, nämlich Linked Data, Web of Data und Semantic Web zusammenhängen bzw. man ist sich nicht einig darüber, welche Kriterien Applikationen erfüllen müssen um den Titel „Semantic Web“ tragen zu dürfen.

Tim Berners-Lee sieht in seinem ersten wegweisenden Paper zu diesem Thema [Ber98], die Umsetzung des Semantic Web als ein Web of Data – ein Web in dem anstatt von Dokumenten Daten verlinkt werden. Diese erste, sehr lose Definition vom Semantic Web macht keine Aussagen über die notwendigerweise zu verwendende Technologie. Sie besagt lediglich, dass das Web als Informationsmedium zu verwenden ist um das Ziel, Daten zu verlinken, umzusetzen. Genau dieser Punkt, nämlich die zu verwendende Technologie, stellt den wesentlichen Streitpunkt in diversen Blogs, wie [Hen09] oder [Hea09] dar. Man ist sich nicht einig darüber, ob Semantic Web an URIs, RDF(S)[7] und HTTP gebunden ist. Laut vorherrschende Meinung des W3C, umgesetzt im Linked Open Data Project, ist das Semantic Web an diese drei Basistechnologien gebunden; Web Applikationen, die sich nicht daran halten, sollten unter einem anderen Label „vermarktet“ werden.
In [Hea09] wird argumentiert, dass im Semantic Web das „Web“ eng mit dem HTTP Protokoll einher geht. Somit kann eine Applikation die keine dereferenzierbaren HTTP

[7] inklusive owl:sameAs und owl:[Inverse]FunctionalProperty

URIs verwendet auch keine Web Applikation sein. Ob RDF(S) oder OWL, oder beide als Ontologiesprache verwendet werden, hängt von der Semantic Web Applikation ab - zulässig sind alle drei Möglichkeiten.

Tim Berners-Lee verdeutlicht in [Ber08] wie er den Zusammenhang zwischen Linked Data und Semantic Web sieht: „Linked Data is Semantic Web done right".

[Hea09] erklärt die Begriffe „Linked Data" und „Web of Data" auf folgende Weise:
„Think about HTML documents; when people started weaving these together with hyperlinks we got a Web of documents. Now think about data. When people started weaving individual bits of data together with RDF triples (that expressed the relationship between these bits of data) we saw the emergence of a Web of data. Linked Data is no more complex than this – connecting related data across the Web using URIs, HTTP and RDF. … So if we link data together using Web technologies … the result is a Web of data."
Somit kann Linked Data als eine konkrete Semantic Web Applikation von der Linked Data Community gesehen werden, die sich das Ziel gesetzt hat, es Person zu ermöglichen strukturierte Daten genau so einfach gemeinsam nutzbar zu machen wie es bereits mit Dokumenten heute der Fall ist [BCH07]. Das Ergebnis des Linked Data Project ist das Web of Data. Dieses setzt nach [Ber06a] die Einhaltung folgender vier Prinzipien voraus:

" - Use URIs as names for things.
- Use HTTP URIs so that people can look up those names.
- When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL).
- Include links to other URIs. So that they can discover more things."

[Ber06a] und [BCH07] sehen in der Einhaltung dieser vier Prinzipien die Möglichkeit einen Mehrwert zu erzeugen, welcher durch die Verlinkung von Daten und Wiederverwendung dieser entsteht. Werden diese Prinzipien nicht eingehalten, wird das Web of Data nicht zerstört, es wird lediglich die Möglichkeit Daten zu verlinken nicht genutzt. Somit können diese vier Prinzipien als die Definition des Web of Data gesehen werden [Hea09].

Im weiteren Verlauf dieser Diplomarbeit werden „Web of Data“ und „Linked Data“ synonym verwendet.

Da das in Teil B dieser Arbeit vorgestellte Rollenmodell im Kontext von Linked Data erstellt wird, wird im Folgenden nur auf Technologien eingegangen die mit dem Web of Data einhergehen.

2.5 Linked Data Technologien

Linked Data würde ohne sorgfältig entwickelte und übereinkommende Standards nicht existieren können, genau so wenig wie das Web nicht ohne HTTP, HTML und XML existieren könnte [SHB06]. Die wesentlichen Technologien und Standards vom Web of Data, die sich durch die vier Prinzipien in [Ber06a] ergeben, sind: URI, HTTP und RDF(S). Um das Web of Data für den Benutzer zugänglich zu machen wird ein Linked Data Browser benötigt, genau so wie im Web ein HTML Browser benötigt wird um Dokumente zu repräsentieren. Dem Linked Data Browser Tabulator wird ein eigener Abschnitt (Teil C) gewidmet, da dieser auch mit dem Ziel Rollen zu interpretieren und dementsprechend zu repräsentieren adaptiert worden ist.

2.5.1 Ressourcen

Im Web of Data werden Dinge, so genannte „items of interest“, beschrieben. Von diesen Dingen werden die relevanten Eigenschaften und Beziehungen zu anderen Dingen beschrieben. In der Web Architektur Terminologie nennt man diese Dinge Ressourcen (resources). [BCH07]
[Wan07] definiert Ressourcen wie folgt: „…resources can be further defined as the abstract entities that have dereferencable URIs”.

Im Web of Data werden alle Informationen über Ressourcen in Form von RDF-Statements ausgedrückt [SCV07]. Jede Ressource wird eindeutig über einen Unified Resource Identifier (URI) identifiziert. Genauer gesagt werden im Web of Data HTTP URIs verwendet um diese im Web dereferenzierbar zu machen. [BCH07]
Im Web sind Dokumente schon immer mittels eines Uniform Resource Locator (URL) identifiziert worden. URIs und URLs teilen dieselbe Syntax, mehr noch, jede URL ist ebenso eine URI. Einerseits ist das gut, weil man somit einfach Statements über Web

Seiten erzeugen kann, andererseits bringt es Probleme mit sich, weil URIs für Dokumente und URIs für Dinge vermischt werden. [SCV07]
Daher unterscheidet man zwischen zwei verschiedenen Ressourcen. Unter dem Aspekt der Dereferenzierbarkeit von HTTP URIs unterscheidet man zwischen Informationsressourcen (Dokumente) und Nicht-Informationsressourcen (Dinge) [BCH07]. Unter dereferenzieren (dereferencing) versteht [BCH07] den Prozess des Nachschlagens einer URI im Web, mit dem Ziel Informationen über die referenzierte Ressource zu erhalten.

2.5.1.1 Informationsressourcen

Alle Ressourcen die wir im dokumentenbasierten Web vorfinden, beispielsweise Dokumente, Bilder oder andere Mediendateien, sind Informationsressourcen. Im Web of Data ist eine Informationsressource beispielsweise ein Dokument in dem Nicht-Informationsressourcen mit ihren Eigenschaften und Verlinkungen mittels RDF Statements beschrieben werden. Informationsressourcen zeichnen sich dadurch aus, dass sie dargestellt (represented) werden können. [BCH07] definiert eine Darstellung wie folgt:
„A representation is a stream of bytes in a certain format, such as HTML, RDF/XML, or JPEG.“

Eine einzige Informationsressource kann viele verschiedene Repräsentation haben, die sich zum Beispiel bezüglich des Formats oder der verwendeten natürlichen Sprache unterscheiden. Wird die URI einer Informationsressource am Server angefragt, so liefert dieser eine Repräsentation der Ressource mit dem Antwortcode „200 OK“ an den Client zurück. Gängige HTML Browser, wie etwa Firefox, stellen RDF Repräsentationen normalerweise im RDF Rohformat dar. Damit kann allerdings der durchschnittliche Benutzer nicht viel anfangen. Zweckmäßiger wäre eine HTML Repräsentation zusätzlich zur RDF Repräsentation. Um dies zu erreichen, kann man sich des HTML Mechanismus, „content negotiation“ bedienen. Dabei wird dem Server im HTML Header mitgeteilt welche Repräsentation der Daten bevorzugt wird. In Abbildung 2 wird die URI <http://example.org/foaf.rdf> dereferenziert. Als akzeptiertes Repräsentationsformat dieser Informationsressource wird „text/html“ definiert. Der Server verfügt über diese Repräsentation dieser Informationsressource und schickt sie mit dem Antwortcode „200 OK“ an den Client. [SCV07] [BCH07] [HTTP99] [WaOl09]

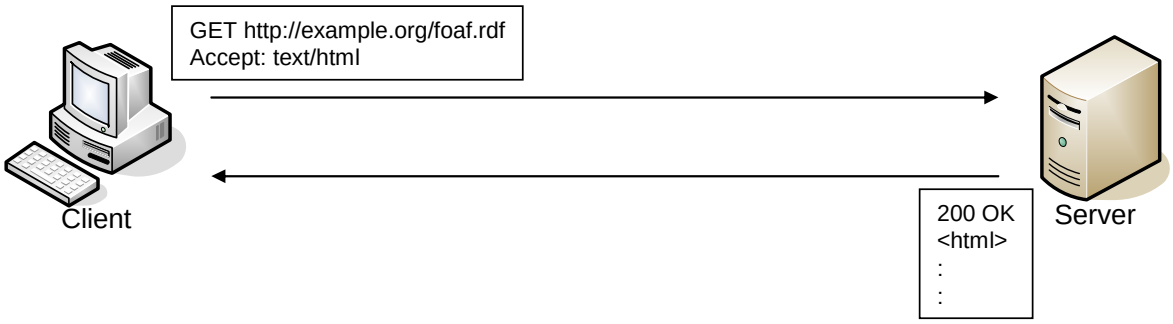


Abbildung 2: Content Negotiation am Beispiel einer URI welche eine Informationsressource identifiziert [BCH07] [HTTP99]

2.5.1.2 Nicht-Informationsressourcen

Aus Sicht der Dereferenzierbarkeit von URIs ist eine Nicht-Informationsressource eine Ressource, die nicht direkt dereferenziert werden kann [BCH07]. Aus einer anderen Perspektive ist eine Nicht-Informationsressource entweder ein Ding, in der Regel ein Ding aus der realen Welt, beispielsweise eine Person, eine Organisation oder ein Gegenstand, oder aber auch eine abstrakte Ideen wie etwa ein Einhorn oder ein Konzept. Alle Dinge, die im Interesse des Autors einer Informationsressource sind, also von denen dieser eine Beschreibung anfertigen möchte, werden als Nicht-Informationsressource bezeichnet. [SCV07] [BCH07] [Boo06]

Bei der Vergabe von URIs ist darauf zu achten, dass diese nicht gleichzeitig eine Informationsressource und eine Nicht-Informationsressource identifiziert. Dies würde bei der Dereferenzierung zu erheblichen Problemen führen. [Boo06] [BFM05]

Nach [SCV07], [BCH07], [WaOl09], [Leb07] und [BePh08] gibt es zwei verschiedene Ansätze wie man URIs vergeben und gleichzeitig das oben erwähnte Problem vermeiden kann.

Der erste Ansatz ist so genannte Hash-URIs für Nicht-Informationsressourcen zu vergeben. Hash-URIs sind URIs, die ein Fragment beinhalten. Ein Fragment wird vom Rest der URI durch das Zeichen „#“ getrennt, beispielsweise http://www.example.org/foaf.rdf#me ist eine URI mit dem Fragment Identifier „#me“.

Wird eine Hash-URI von einem Client beim Server angefragt, so ist es notwendig zuvor den Fragmentteil der URI abzuschneiden. Das bedeutet, eine Hash-URI kann nicht direkt abgerufen werden und kann somit kein Web Dokument, also keine Informationsressource

identifizieren. Der weitere Dereferenziervorgang entspricht dann dem einer Informationsressource (vgl. Abbildung 2).

Der zweite Ansatz besteht darin so genannte „303 URIs“ für Nicht-Informationsressourcen zu vergeben um diese von Informationsressourcen zu unterscheiden. Dazu wird der spezielle Statuscode von HTTP„303 See Other“ verwendet. Eine 303 URI hat kein Fragment und unterscheidet sich von der Syntax her nicht von URIs die Dokumente identifizieren.
Wird eine 303 URI beim Server angefragt, antwortet der Server mit dem Statuscode „303 See Other“ und liefert zusätzlich noch jene URI mit, mittels welcher der Server Informationen über die eben angefragte Nicht-Informationsressource liefern kann. Wie in Abbildung 3 dargestellt, fragt der Client beim Server die URI http://www.example.org/id/hans an. Der Server antwortet mit dem Statuscode „303 See Other“ und liefert im Header unter „Location“ die URI der Informationsressource mit. Daraufhin fragt der Client erneut beim Server an, jedoch diesmal mit der vom Server zurückgegebenen Adresse der Informationsressource. Der Server antwortet nun mit „200 OK“ und liefert die Repräsentation der Informationsressource mit.

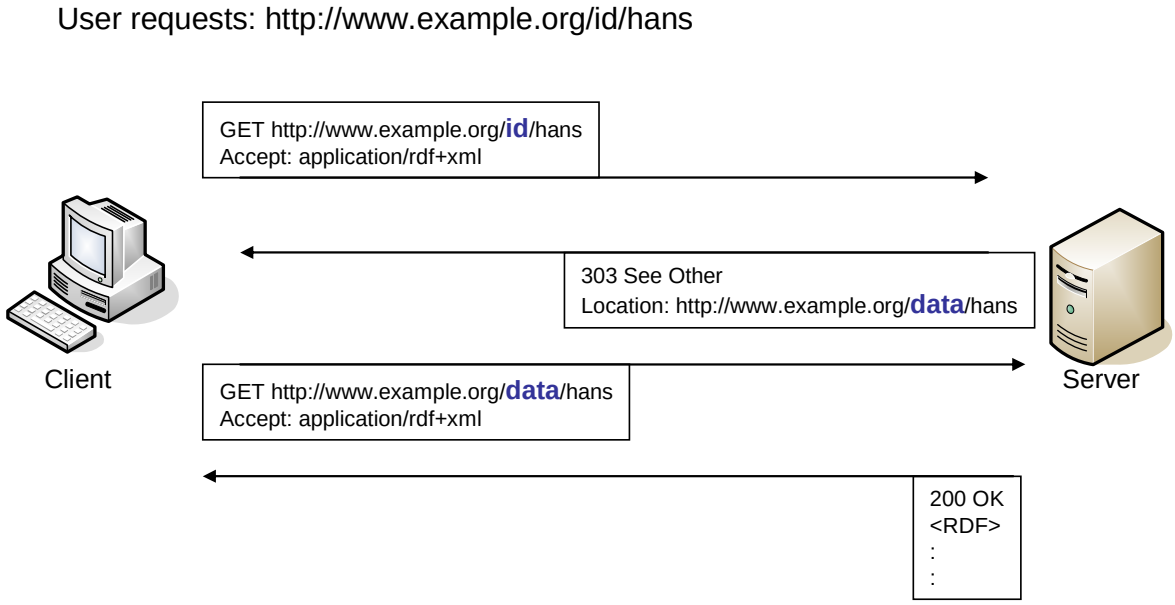


Abbildung 3: Anfrage einer 303 URI [BCH07] [HTTP99]

Die Entscheidung welchen Ansatz man verwendet, hängt von mehreren Faktoren ab:
Hash-URIs haben den Vorteil, dass quantitativ weniger Kommunikationsaufwand mit dem Server aufkommt, da pro Anfrage einer URI eine einzigen Antwort vom Server genügt. Mehr noch, dadurch dass URIs den nicht-hash Teil einer URI teilen können, kann es vorkommen, dass mit einem Aufruf andere Aufrufe überflüssig werden. Beispielsweise die Repräsentation der URIs http://www.example.org/foaf.rdf#me, http://www.example.org/foaf.rdf#hans, und http://www.example.org/foaf.rdf#marie kann mit einer einzigen Anfrage am Server erhalten werden. Dies ist allerdings ein zweischneidiges Schwert, denn wenn man sich nur für eine der URIs interessiert, wird trotz allem die gesamte Informationsressource geladen und somit auch die Informationen zu URIs, die vielleicht nicht von Interesse sind. [SCV07] [BCH07]
Verwendet man 303 URIs so ist man flexibler. Es kann für jede URI ein separater Verweis auf eine Informationsressource erstellt werden. Dadurch ist es sowohl möglich eine Informationsressource pro 303 URI zu definieren, als auch eine einzige große Informationsressource für mehrere 303 URIs, oder irgendwas zwischen diesen beiden Extrema. Der Nachteil dieses Ansatzes ist der höhere Kommunikationsaufwand mit dem Server. [SCV07] [BCH07]
Aufgrund der Vor- und Nachteile der beiden Ansätze ergibt sich für Hash-URIs ein bevorzugtes Anwendungsgebiet bei kleineren und stabilen Mengen an Ressourcen, die gemeinsam entwickelt werden. Beispielsweise ist das der Fall bei RDF Schema, Vokabularen und OWL Ontologien, da einerseits die beschriebenen Ressourcen mit hoher Wahrscheinlichkeit gemeinsam genutzt werden und es somit Sinn macht ein einziges Mal die gesamte Informationsressource zu laden und da es andererseits eher unwahrscheinlich ist, dass die Anzahl der Ressourcen zukünftig extrem wachsen wird. Hash-URIs sind auch bei quick-and-dirty RDF Publikationen beliebt, wo einfach statische RDF Dateien auf einen Webserver gelegt werden ohne dem Konsumenten der Daten die Möglichkeit einer Content Negotiation zu bieten. Im Zweifelsfalle sollten 303 URIs verwendet werden, insbesondere bei großen Datenmengen. [SCV07] [BCH07]

2.5.2 Uniform Resource Identifier (URI)

In [BFM05] ist ein Uniform Resource Identifier (URI) definiert, als „… a compact sequence of characters that identifies an abstract or physical resource.“. Da im Web of Data URIs dereferenzierbar sein sollen und dafür das Protokoll HTTP verwendet wird (vgl. Kaptitel 2.4), spricht man im Web of Data von HTTP URIs, die entweder eine Nicht-

Informationsressource (abstract resource) oder eine Informationsressource (physical resource) identifizieren [Wan07] [Ber03] [Ber07a].

URIs nehmen eine sehr zentrale Stellung im Web of Data ein. Mittels RDF(S) werden URIs (als Namen für Ressourcen) mit Hilfe von URIs verlinkt. Dementsprechend „gut" sollen die Namen von Autoren gewählt werden um es anderen Benutzern einfacher bzw. möglich zu machen auf andere Ressourcen zu verweisen. Die Namensgebung für URIs soll möglichst sprechend sein, demnach sind kurze mnemonische Namen besser, als lange kryptische [BCH07].

Wie schon oftmals erwähnt und äußerst wichtig, beschränkt man sich im Web of Data auf HTTP URIs mit der Absicht, dass URIs im Web dereferenzierbar sein sollen und somit eine Repräsentation an den Anfrager zurück geliefert werden kann. D.h., es sollen für die gesamte Namensgebung von Ressourcen HTTP URIs verwendet werden, da es das am weitesten verbreitete URI Schema ist und von den meisten Tools und Infrastrukturen unterstützt wird.

Sehr wesentlich für den Erfolg vom Web of Data ist, dass URIs stabil und persistent bleiben. Das Web of Data lebt von Verlinkungen und jede nachträgliche Änderung von bereits verlinkten URIs würde die Verlinkung brechen. [BCH07]

2.5.2.1 Zusammenhang zwischen Ressource und URI

Im Web of Data entspricht eine Nicht-Informationsressource einem Objekt aus der realen Welt (bzw. einer Idee, einem Konzept, …). Eine Nicht-Informationsressource wird durch eine oder mehrere URIs identifiziert (vgl. Abbildung 4). Dies ist die Intension von [BHL01]. Wenn möglich soll es vermieden werden, dass mehrere URIs eine Nicht-Informationsressource identifizieren, da sich dies negativ auf den Netzwerkeffekt auswirkt [JaWa04].

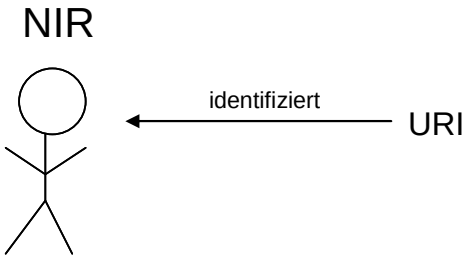


Abbildung 4: Zusammenhang zwischen NIR und URI lt. [BHL01]

Während nach [BHL01], [Ber03] und [Wan07] eine URI eine Ressource (Nicht-Informationsressource oder Informationsressource) identifiziert, sieht [Boo06] dies anders. [Boo06] meint, dass eine URI eine Nicht-Informationsressource nur benennt, aber nicht automatisch identifiziert, in dem Sinne, dass eine URI ein eindeutiger Name für ein Ding ist. Ob nun eine URI eine Ressource nur benennt oder identifiziert, ist eine philosophische Frage und im Rahmen dieser Diplomarbeit nicht weiter von Relevanz. Von Bedeutung ist allerdings seine weitere Erläuterung, die besagt, dass durch die Beschreibung dieses Dings bzw. der Nicht-Informationsressource ein Modell als Stellvertreter für die Nicht-Informationsressource entsteht, welches manipuliert werden kann (vgl. Abbildung 5).
Als Beispiel wird die Nicht-Informationsressource Mr. Black im Web beschrieben. Diese Nicht-Informationsressource wird mittels einer URI benannt/identifiziert. Weiters wird mittels RDF(S) mit dieser URI eine Beschreibung der Ressource Mr. Black erstellt, wodurch ein Modell als Stellvertreter von Mr. Black entsteht.

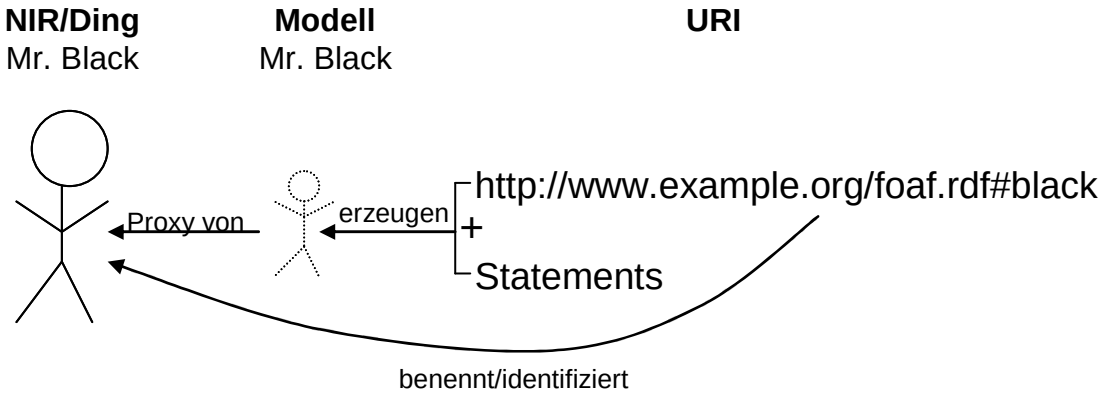


Abbildung 5: Zusammenhang zwischen URI und Nicht-Informationsressource (NIR)

[Boo06] zeigt mit dieser Auffassung des Zusammenhangs zwischen Nicht-Informationsressource und URI auf, dass URIs mit ihrer Beschreibung kontextspezifische Modelle erzeugen und somit die URI selbst auch kontextspezifisch ist. Wenn es beispielsweis zwei URIs gibt mit welchen Mr. Black als Angestellter beschrieben wird, einmal bei der CompZ und einmal bei der UniXY, macht es einen Unterschied welche URI verwendet wird. Beide URIs identifizieren dieselbe Person, nämlich Mr. Black, allerdings in unterschiedlichen Kontexten. Würde man nur mit einer URI eine ganzheitliche Beschreibung von Mr. Black erstellen so kann man nicht mehr feststellen in welcher Anstellung welche Attribute Gültigkeit besitzen (vgl. Kapitel 2.6.1).

2.5.2.2 Autoritative und nicht-autoritative Beschreibungen

Je nachdem, ob der Namensraum einer URI, welche eine Nicht-Informationsressource identifiziert, mit dem Namensraum der URI der Informationsressource in der diese beschrieben wird übereinstimmt, spricht man von einer autoritativen bzw. nicht-autoritativen Beschreibung einer Nicht-Informationsressource [BCH07]. Wenn beispielsweise Mr. Black, der durch die URI http://example.org/foaf.rdf#black identifiziert wird in der Informationsressource http://example.org/foaf.rdf beschrieben wird, dann spricht man von einer autoritativen Beschreibung. Wird Mr. Black mit seiner URI in einer anderen Informationsressource beschrieben, so spricht man von einer nicht-autoritativen Beschreibung.

Es ist wesentlich, dass URIs in einem Namensraum definiert werden, der unter eigener Kontrolle ist, wie dies bei einer autoritativen Beschreibung der Fall ist, damit man die Möglichkeit besitzt, sie dereferenzierbar zu machen. Wird eine Nicht-Informationsressource in einer Informationsressource nicht-autoritativ beschrieben, so kann auf diese Beschreibung nicht direkt verwiesen werden. Denn eine Dereferenzierung der URI liefert nur die autoritative Beschreibung der Ressource und über diese hat man in der Regel als nicht-autoritativer Autor keine Kontrolle. Die einzige Möglichkeit auf eine nicht-autoritative Beschreibung zu verweisen ist, wenn der Autor der autoritativen Beschreibung von der nicht-autoritativen Beschreibung weiß und auf diese mittels eines rdfs:seeAlso Links verweist. Mittels dieses Links wird ausgedrückt, dass in der Objekt-URI weitere Informationen über die Subjekt-URI gefunden werden können [MaMi04b]. Darauf wird zu einem späteren Zeitpunkt noch genauer eingegangen.

Dadurch, dass auf nicht-autoritative Beschreibungen nicht direkt verwiesen werden kann, schränken sie die Navigierbarkeit stark ein.

2.5.3 Resource Description Framework (RDF)

Das Resource Description Framework (RDF) ist eine Sprache zum Abbilden von strukturierten Informationen über Ressourcen im World Wide Web [MaMi04a]. RDF macht es uns möglich Ressourcen zu beschreiben und ist somit die Grundlage für das Web of Data [BCH07]. Es bietet ein graphen-orientiertes Datenmodell, welches sehr einfach ist und genau auf die Web Architektur zugeschnitten ist [BCH07]. RDF erlaubt es im Gegensatz zu XML neben der Strukturierung der Daten auch Aussagen darüber zu machen, was diese Struktur bedeutet [BHL01] [DMH+00]. D.h., es ist lt. [BlPe06]

möglich, eine Metadaten-Infrastruktur aufzubauen. Daher wird RDF als grundlegendes Darstellungsformat für die Entwicklung des Web of Data angesehen.
Der Kleber, der das traditionelle Web zusammenhält, sind Hyperlinks zwischen HTML Seiten. Der Kleber, der das Web of Data zusammenhält, sind RDF Links. Ein RDF Link besagt: „one piece of data has some kind of relationship to another piece of data“. [BCH07]

„Ein RDF-Dokument beschreibt einen gerichteten Graphen, also eine Menge von Knoten, die durch gerichtete Kanten (”Pfeile“) verbunden werden. Dabei sind sowohl Knoten als auch Kanten mit eindeutigen Bezeichnern beschriftet.“ [HKRS08] In Abbildung 6 wird ein RDF Graph gezeigt, der ausdrückt, dass Peter Black, Tina White kennt. Ein RDF Graph, bestehend aus einem Subjekt, einem Prädikat und einem Objekt, wird auch Tripple genannt. Das Subjekt des Tripples ist die URI, welche die zu beschreibende Ressource benennt, in unserem Beispiel Peter Black. Das Objekt kann entweder ein Literal, ein String, eine Nummer oder ein Datum sein, oder eine URI einer anderen Ressource, welche in Beziehung zur Objektressource steht. In Abbildung 6 wird das Objekt Tina White durch die URI http://huemer.lstadler.net/foaf.rdf#white identifiziert. Das Prädikat in der Mitte eines Tripples drückt die Art der Beziehung zwischen dem Subjekt und dem Objekt aus und ist ebenso wie das Subjekt immer eine URI. Prädikat URIs kommen oft von RDF-Vokabularen, wie auch aus Prädikat http://xmlns.com/foaf/0.1/knows im Beispiel in Abbildung 6.
Vokabulare sind eine Sammlung von URIs die verwendet werden können um Informationen über ein bestimmtes Gebiet abzubilden. Ein Beispiel für ein Vokabular ist das FOAF-Vokabular[8], wobei FOAF für „Fried of a Friend“ steht. Es wird dazu verwendet um Menschen und ihre Beziehungen untereinander zu beschreiben. [HKRS08] [BCH07] [MaMi04a] [BrMi07] [AnHa08] [KBM08]

[8] http://xmlns.com/foaf/spec/

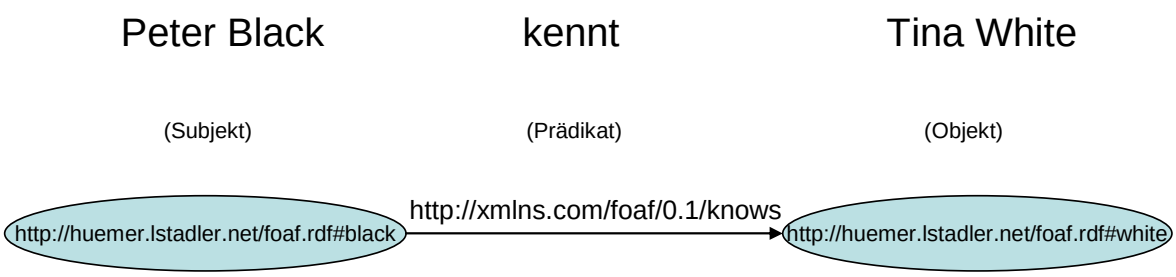


Abbildung 6: RDF-Tripple

Abhängig von der Art des Objektes unterscheidet man zwischen Literal Tripple und RDF Links. Literal Tripple haben als Objekt ein Literal, beispielsweise einen String, eine Nummer oder ein Datum. Literale werden benutzt um Eigenschaften von Ressourcen zu beschreiben, beispielsweise den Namen oder das Geburtsdatum einer Person. RDF Links sind typisierte Links zwischen zwei Ressourcen, d.h. sowohl das Objekt als auch das Subjekt sind eine URI. Die Prädikat URI definiert den Typ des Links, beispielsweise eine „kennt“ Beziehung wie in Abbildung 6. [HKRS08] [MaMi04a]

Es gibt verschiedene Syntaxen um Informationen im Web of Data zu serialisieren und auszutauschen. 1998 hat Tim Berners-Lee Notation 3 (N3) [Ber06b] [N3P00] vorgeschlagen. Dies ist eine Syntax die auch komplexere Ausdrücke wie Regeln und Pfade beinhaltet. Später hat das W3C in [GrBe04] einen weniger komplexen Teil von N3 unter dem Namen N-Triples empfohlen. Auch diese Syntax ist noch einmal überarbeitet worden zum Turtle Standard [BeBe08].
Aufgrund der besseren Unterstützung von Programmierbibliotheken ist die am häufigsten verwendete Syntax die RDF/XML Syntax. Diese Syntax ist zwar für den Mensch schwer lesbar, jedoch ist sie aufgrund der weiten Verbreitung von XML Parsern von Maschinen leichter einzulesen und somit auch bevorzugtes Serialisierungsformat. [HKRS08]

2.5.4 Resource Description Framework Schema (RDFS)

„Mithilfe von RDF Schema (kurz RDFS), einem weiteren Bestandteil der RDF Recommendation des W3C, …, ist es möglich, … so genanntes terminologisches Wissen oder auch Schemawissen … zu spezifizieren“ [HKRS08, S. 67]. Terminologisches Wissen oder Schemawissen sind Hintergrundinformationen über ein Vokabular, welches notwendig ist, um für Maschinen die Zusammenhänge zwischen Vokabeln zu definieren. Bei der Erstellung oder Verwendung eines Vokabulars ist es nahe liegend, dass beispielsweise nur Personen verheiratet sein können oder dass jede Person einen Vater und

eine Mutter hat. Für Maschinen sind Vokabeln nur Zeichenketten, die ohne Hintergrundinformation sehr wenig Semantik besitzen. [HKRS08] [AnHa08] [KBM08] [MaMi04b]

RDFS selbst ist zunächst nichts anderes als ein spezielles RDF-Vokabular. Ein RDF-Vokabular, das es ermöglicht „innerhalb eines RDFS-Dokuments Aussagen über die semantische Beziehungen der Termini eines beliebigen nutzerdefinierten Vokabulars zu machen." [HKRS08, S. 67]. Mit RDFS kann ein neues themenspezifisches Vokabular, wie beispielsweise das FOAF-Vokabular[9], definiert werden und ein Teil der Semantik im Dokument spezifiziert werden. RDFS wird als leichtgewichtige Ontologiesprache bezeichnet, da die Ausdrucksmöglichkeiten sehr eingeschränkt sind. Jedoch wie [Hen09] bemerkt hat: „A little semantic goes a long way." Wie bereits erwähnt, kann man sich bei Bedarf auch mit einer ausdrucksstärkeren Sprache wie OWL behelfen.
Im Folgenden wird auf einige Grundlagen von RDF(S) eingegangen:

2.5.4.1 Klassifizieren mit RDF(S)

In RDF verwendet man das Prädikat rdf:type um Instanzen einer Klasse zugehörig zu machen, d.h. um eine Ressource (mit einer URI als Identifier) zu klassifizieren. Eine URI kann aber auch durch die Verwendung eines Prädikates einer Klasse zugehörig gemacht werden, nämlich dann, wenn zu einem Prädikat, mittels rdfs:domain, ausgedrückt wird, dass jede Instanz, die diese Eigenschaft hat, einer gewissen Klasse zugehörig ist. Weiters gibt es mit dem Prädikat rdfs:range die Möglichkeit auszudrücken, dass jeder Wert eines Prädikates einer Klasse angehört. [MaMi04a] [HKRS08]
Z.B.:
Das Tripple

```
ex:black rdf:type foaf:Person.
```

besagt, dass die Ressource, die mit ex:black benannt ist eine Instanz von der Klasse foaf:Person ist.

Die Tripple

```
foaf:name rdfs:domain foaf:Person.
ex:black foaf:name "Peter Black".
```

[9] http://xmlns.com/foaf/spec/

besagen, dass jede URI, die als Attribut foaf:name hat, zur Klasse foaf:Person gehört. Die URI ex:black hat einen foaf:name mit dem Wert „Peter Black“ und gehört folge dessen zur Klasse foaf:Person bzw. ist als foaf:Person klassifiziert.

Die Tripple

```
foaf:knows rdfs:range foaf:Person.
ex:black foaf:knows ex:white.
```

besagen, dass jeder Wert (Objekt) vom Prädikat foaf:knows der Klasse foaf:Person angehört. Die URI ex:white ist der Wert (Objekt) des Prädikates foaf:knows und folge dessen wird ex:white als foaf:Person klassifiziert.

Wie gezeigt worden ist, kann eine Instanz einer Klasse in RDF(S) unabhängig von ihrer Klassifizierung Eigenschaften annehmen und auch zu unterschiedlichen Klassen gehören, im Gegensatz zur objekt-orientierten Welt. In der objekt-orientierten Welt gehört eine Instanz meist genau einer Klasse an und kann nur Eigenschaften haben welche in der Klasse definiert sind.

2.5.4.2 Klassenhierarchien mit RDF(S)

In RDF(S) können mittels dem Attribut rdfs:subClassOf Klassenhierarchien gebildet werden. Nachdem es in RDF(S) keine Instanzmethoden gibt und Klassen nicht vordefinierte Eigenschaften in Form von Attributen haben, gibt es auch keine Strukturvererbung, im Gegensatz zur objekt-orientierten Welt. In der objekt-orientierten Welt können in Klassen Instanzmethoden beschrieben werden, welche auf alle Instanzen der Klasse und ihrer Subklassen angewandt werden können. Außerdem können in Klassen Instanzattribute definiert werden, welche alle Instanzen dieser Klasse und ihrer Subklassen haben. In RDF(S) gibt es keine Methoden.

In RDF(S) ist eine Subklassenbeziehung zwischen zwei Klassen transitiv und besagt, dass alle Instanzen der einen Klasse Instanzen der anderen Klasse sind [MaMi04b].

Z.B.:

Die Tripple

```
foaf:Person rdfs:subClassOf foaf:Agent.
ex:black rdf:type foaf:Person.
```

besagen, dass ex:black zur Klasse foaf:Person gehört und aufgrund der Subklassenbeziehung zwischen foaf:Person und foaf:Agent gehört ex:black auch zur Klasse foaf:Agent.

2.5.4.3 Prädikathierarchien mit RDF(S)

In RDFS können auch Prädikathierarchien erstellt werden, in der zwischen den Attributen eine Spezialisierungsbeziehung besteht. Prädikathierarchien werden verwendet um auszudrücken, dass alle Ressourcen, welche durch das eine Prädikat in Beziehung gesetzt sind, auch durch das andere in Beziehung gesetzt sind [MaMi04b].

Z.B.:

Die Tripple

```
ex:isHappyMarriedWith rdfs:subPropertyOf ex:isMarriedWith.
ex:black ex:isHappyMarriedWith ex:white.
```

besagen, dass ex:black nicht nur glücklich mit ex:white verheiratet ist, sondern auch dass dieser mit ihr verheiratet ist.

2.6 URIs und Identität – das Coreferenz-Problem

Da das Web of Data dezentral ist und es viele Autoren von Informationsressourcen gibt, wäre es naiv anzunehmen, dass immer nur eine einzige Beschreibung eines Objektes aus der realen Welt existiert. Es können von einer Entität mehrere voneinander unabhängige Beschreibungen existieren, quasi mehrere unabhängige Modelle der Entität in einem möglicherweise unterschiedlichen Kontext, was von [JGM07] als das Coreferenz-Problem bezeichnet wird. Beispielsweise vergeben DBpedia[10], Geonames[11], CIA Factbook[12] und Eurostat[13] jeweils eine unterschiedliche URI für dasselbe Land. Oder auch Citeseer[14], DBLP[15], IEEE[16] und ACM[17] vergeben unterschiedliche URIs für dieselben Autoren und Artikel. Der Grund für die separate Vergabe von URIs liegt darin, dass die Informationsprovider Kontrolle über die URIs haben wollen und dies geht nur, wenn sie die Kontrolle über den Namensraum in dem die URI definiert ist, haben (vgl. Kapitel 2.5.2.2).

Dieses Coreferenz-Problem, also wenn eine Nicht-Informationsressource durch mehrere URIs identifiziert wird, stellt im Web of Data eine große Herausforderung dar. Denn hat man zwei textuell unterschiedliche URIs, so weiß man nicht ob diese dieselbe Nicht-Informationsressource identifizieren oder nicht, da im Web of Data die Unique Name

[10] http://wiki.dbpedia.org
[11] http://www.geonames.org/
[12] https://www.cia.gov/library/publications/the-world-factbook/
[13] http://epp.eurostat.ec.europa.eu/portal/page/portal/eurostat/home/
[14] http://citeseer.ist.psu.edu/
[15] http://www.informatik.uni-trier.de/~ley/db/
[16] http://www.ieee.org/portal/site
[17] http://www.acm.org/

Assumption nicht gilt. Die Unique Name Assumption besagt, dass zwei Objekte mit unterschiedlichen Namen unterschiedliche Identität besitzen.
Einerseits ist es schwierig festzustellen ob zwei textuell unterschiedliche URIs dieselbe Nicht-Informationsressource identifizieren, anderseits stellt sich die Frage wie man damit umgeht wenn man festgestellt hat, dass zwei URIs dieselbe Nicht-Informationsressource identifizieren. Auf letzteres, nämlich auf dem Umgang mit dem Coreferenz-Problem, soll nun näher eingegangen werden.

2.6.1 Empfehlung der Linked Data Community

Die Linked Data Community empfiehlt URIs gleicher Identität mittels owl:sameAs zu verlinken [ESW09]. D.h. wenn zwei URIs dieselbe Nicht-Informationsressource identifizieren, dann sollen diese mit owl:sameAs verknüpft werden. Die damit verknüpften URIs (das Subjekt und Objekt des Statements) werden zu alias URIs und können äquivalent für die Beschreibung des Objektes aus der realen Welt verwendet werden [BCH07] [DeSc04] (vgl. Abbildung 7). Wie bereits in Kapitel 2.5.2.1 gezeigt, gibt es bei der Interpretation des Zusammenhangs zwischen URI und Nicht-Informationsressource von Tim Berners-Lee in [BHL01] die „Modell-Ebene“ nicht. Deshalb wird in Abbildung 8 owl:sameAs an die Interpretation des Zusammenhangs zwischen URI und Ressource von [Boo06] angepasst.

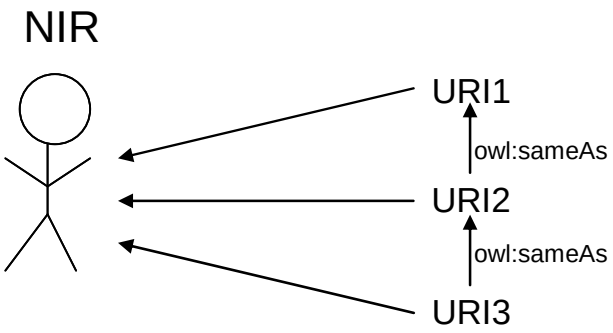


Abbildung 7: Verlinkung von URIs mit owl:sameAs nach [BHL01]

Werden zwei URIs bzw. Modelle mit owl:sameAs verlinkt, so entsteht folge dessen anstatt von zwei Modellen nur ein einziges Modell, welches als Stellvertreter für die Nicht-Informationsressource gilt (vgl. Abbildung 8). Auf diese Art und Weise kann zwar die Aussage getätigt werden, dass sich die Beschreibungen die mittels der URIs getätigt werden auf das selbe Objekt aus der realen Welt beziehen, allerdings geht die Möglichkeit

kontextspezifische Aussagen zu tätigen verloren. D.h., dadurch, dass zwei Modelle zu einem zusammengefügt werden, beschreiben die Summe der Aussagen die mit den URIs getätigt worden sind das fusionierte Modell und es kann nicht mehr modellspezifisch bzw. kontextspezifisch zwischen den Aussagen unterschieden werden. Dies führt zu einem erheblichen Informationsverlust.

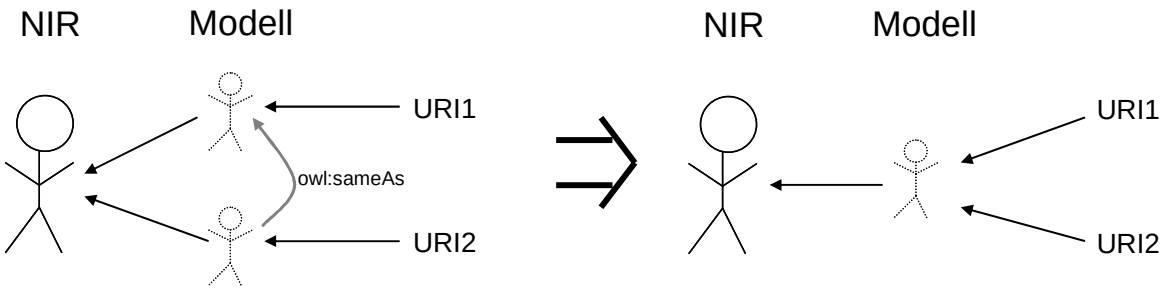


Abbildung 8: Verlinkung von URIs mit owl:sameAs angepasst an Aussagen von [Boo06]

Das Prädikat owl:sameAs stellt im Web of Data die einzige Möglichkeit dar, auszudrücken, dass zwei URIs dasselbe Objekt aus der realen Welt identifizieren [DeSc04].

2.6.2 Ansatz nach [JGM07] und [JGM08]

Die Autoren dieses Ansatzes befinden die Empfehlung der Linked Data Community in vielen Situationen für unpassend. Beispielsweise, wenn URIs zwar dasselbe Objekt aus der realen Welt identifizieren, die Beschreibungen sich jedoch im Kontext unterscheiden. Durch eine Verlinkung mittels owl:sameAs geht der Kontext jeder einzelnen Beschreibung verloren, was zu einem Informationsverlust führt. D.h. die Semantik von owl:sameAs ist zu mächtig und andere Alternativen, wie beispielsweise rdfs:seeAlso erfüllen nicht den Zweck.

Um Coreferenzen zwischen den vielen URIs im Web zu verwalten, haben [JGM07] und [JGM08] das Consistent Reference Service (CRS) entwickelt. Dabei bilden sie so genannte Bündel, welche URIs, die dieselbe Ressource benennen aber in einem unterschiedlichen Kontext zueinander stehen, gruppieren. Es besteht auch die Möglichkeit URIs kontextspezifisch und ressourcenspezifisch zu bündeln. Ein ressourcenspezifisches Bündel besteht dann aus mehreren Bündeln, ein Bündel je Kontext.

Das CRS ist ein RDF-Dokument in welchem URIs verlinkt sind, welche dasselbe Objekt aus der realen Welt benennen. In Abbildung 9 ist ein Ausschnitt aus dem durchgängigen

Beispiel aus Kapitel 1.2 dargestellt, indem vom FOAF-Dokument von Peter Black mittels coref:hasCRS auf ein Bündel in einem CRS Dokument verwiesen wird, in welchem die URIs die ebenfalls Peter Black benennen aufgelistet sind. Mit dem Prädikat coref:duplicate werden zu einem Bündel URIs hinzugefügt. Das Prädikat coref:insertedOn gibt die Aktualität des CRS an.

RDF-Dokument: http://huemer.lstadler.net/role/uc2/black.rdf

```
@prefix coref: <http://www.resist.ecs.soton.ac.uk/ontology/coref#>
@prefix x: <http://huemer.lstadler.net/role/uc2/black.rdf#
@prefix foaf: <http://xmlns.com/foaf/0.1/>

x:black a foaf:Person;
                foaf:name „Peter Black“;
                coref:hasCRS <http://www.example.org/crs.rdf#person_123>.
```

CRS – RDF-Dokument: http://www.example.org/crs.rdf

```
@prefix coref: <http://www.resist.ecs.soton.ac.uk/ontology/coref#>
@prefix foaf: <http://xmlns.com/foaf/0.1/>

<http://www.example.org/crs#person_123> a coref:Bundle;
                coref: duplicate <http://huemer.lstadler.net/role/uc2/emp1.rdf#empBlack1>,
                <http://huemer.lstadler.net/role/uc2/emp2.rdf#empBlack2>;
                coref:insertedOn „2009-02-27“.
```

Abbildung 9: Bündelung mehrerer URIs mittels Consistent Reference Service

All jene, die eine gesamtheitliche Beschreibung von Peter Black wollen, brauchen nur der CRS Verlinkung folgen und alle Attribute kumulieren, was der Semantik von owl:sameAs entspricht. All jene, die eine Trennung der Beschreibung wünschen, erhalten auch diese.

Das CRS ist ein dezentraler und verteilter Ansatz, um URIs mit gleicher Identität zu verwalten und dabei die Prinzipien der Web Architektur einhält. Da Bündel aus Bündeln bestehen können, kann damit auch eine hierarchische Struktur nachgebildet werden. Einziger Schwachpunkt dieses Ansatzes ist, dass zwar eine kontextspezifische Trennung von URIs möglich ist, jedoch der Kontext nicht definiert werden kann. Beispielsweise ist es nicht möglich auszudrücken, dass das Bündel <http://www.ex.org/crs#person_123>, URIs, die Peter Black als Angestellten identifizieren, bündelt.

2.6.3 Ansatz nach [BSCT08]

[BSCT08] machen das Fehlen eines globalen Referenz- und Adressierungsmechanismus zum Lokalisieren und Abrufen von Ressourcen, für den ihrer Meinung nach langsamer als erwarteten Fortschrittsverlauf des Verlinkens von Daten verantwortlich. Daher schlagen sie analog zum Domain Name System (DNS), welches im World Wide Web URLs in physikalische Ortsangaben auflöst, für das Semantic Web ein Entity Name System (ENS) vor. Das ENS soll verteilt und dezentral sein, sehr ähnlich wie das DNS.

[BSCT08] beschreiben das Entity Name System, welches im EU-Projekt Okkam[18] erste Prototypen zur Verfügung stellt, als ein Service „which stores and makes available for reuse URIs for any type of entity in a fully decentralized and open knowledge publication space." Mit einem solchen Ansatz wollen sie das rasche Anwachsen von URIs für dasselbe Objekt aus der realen Welt (Coreferenz-Problem) stoppen und das wieder verwenden von bereits existierenden URIs fördern. Sie befürworten einen Ansatz, in dem es nur eine einzige globale URI pro Ressource gibt, welche immer dann, wenn eine Aussage über die Ressource getätigt wird, verwendet werden soll.

Die Basisfunktionalität des ENS beinhaltet das Suchen nach Entitäten, das Hinzufügen von neuen Entitäten und das Erstellen von neuen Identifiern (Bezeichnern). Als Identifier werden absolute URIs (URIs ohne Fragmentidentifier) vorgeschlagen. Als Schlüsselfeature gilt das Suchen nach Entitäten, denn wenn eine Anfrage eingereicht wird, muss entschieden werden ob diese einer bereits existierenden Entität entspricht oder ob eine neue Entität erzeugt werden muss. Dabei geht das ENS so vor, dass eine Liste mit möglichen Kandidaten erstellt wird, welche hinsichtlich der Anfrage gereiht werden. Entspricht keiner der Kandidaten der gefragten Entität, so wird ein neuer Identifier erzeugt.

[JGM08] kritisieren diesen Ansatz stark. Als erstes kritisieren sie die Analogie zum DNS und halten diese für nicht passend, da das DNS ein hierarchisches System zum Auffinden des Ortes eine Ressource ist. Das Semantic Web wiederum benötigt ein System zum Auffinden von identischen Ressourcen und das sind zwei unterschiedliche Aufgaben. Unter anderem kritisieren sie auch die Architektur des ENS. Das ENS ist derzeit als ein zentrales System implementiert, was gegen die Prinzipien des Semantic Webs ist.

Als Kritik kann noch hinzugefügt werden, dass der Kontext einer Ressourcenbeschreibung vernachlässigt wird. Denn, gibt es nur eine globale URI pro Ressource so ist hierfür kein

[18] http://www.okkam.org/

Platz. Wie wichtig der Kontext der Beschreibung einer Ressource ist, wurde bereits in vorangegangen Kapiteln erläutert.

2.6.4 Zusammenfassung

Die Linked Data Community in [ESW09] empfiehlt, dass URIs, die auf dieselbe Nicht-Informationsressource verweisen mittels owl:sameAs verknüpft werden. Damit wird ausgedrückt, dass sie dieselbe Identität teilen und werden zu alias URIs [DeSc04]. Somit werden die Modelle, die durch die Beschreibung einer Nicht-Informationsressource mit den URIs entstehen, zu einem Modell zusammengefasst.
[JGM07] und [JGM08] kritisieren diesen Ansatz, da dadurch eine kontextspezifische Unterscheidung von Beschreibungen verloren geht und lösen das Coreferenz-Problem auf eine Art und Weise, die ausdrückt, dass URIs dieselbe Nicht-Informationsressource identifizieren, jedoch eine kontextspezifische Trennung von Beschreibungen nach wie vor möglich ist.
Die Lösung von [BSCT08] geht in eine andere Richtung und zielt darauf ab, dass eine Ressource nur von einer einzigen URI identifiziert wird.

[Hal06] hat sich mit der Definition von Identität (im Sinne von Gleichheit) auseinander gesetzt. Er zitiert dabei das Leibnizsche Gesetz und definiert Identität wie folgt: “The classic definition of identity is whether or not two objects are in fact, on some given level, the same.”. Das bedeutet, dass die Identität zweier Objekten abhängig ist vom Betrachtungslevel.

Aufgrund der Semantik von owl:sameAs kann somit gefolgert werden, dass sich die Identitätsfrage lt. der Linked Data Community darauf beschränkt ob eine URI dieselbe Nicht-Informationsressource identifiziert oder nicht. D.h. der Level auf dem die Linked Data Community die Identität betrachtet, ist auf Ebene der Entität. Dem Kontext wird keine Beachtung geschenkt. Um owl:sameAs kompatible zum Modell von [Boo06] zu machen (vgl. Abbildung 5), wird owl:sameAs so betrachtet, dass es URIs verknüpft, die sowohl auf Entitätsebene als auch auf Kontextebene identisch sind. (vgl. Kapitel 2.6.1).

[JGM07] und [JGM08] betrachten nicht nur die Entitätsgleichheit, also den Entitätslevel, sondern auch den Kontext der Beschreibung. D.h. sie führen unter dem Entitätslevel noch einen weiteren Level ein, den wir hier Kontextlevel nennen. In Abbildung 10 ist dieser

Sachverhalt verdeutlicht. Vom Entitätslevel aus betrachtet, bzw. von Entität A aus betrachtet, sind alle darunter liegenden Objekte (Modelle, Beschreibungen) identisch. Innerhalb des Kontextlevels können durch kontextspezifische Bündelungen Hierarchien entstehen.

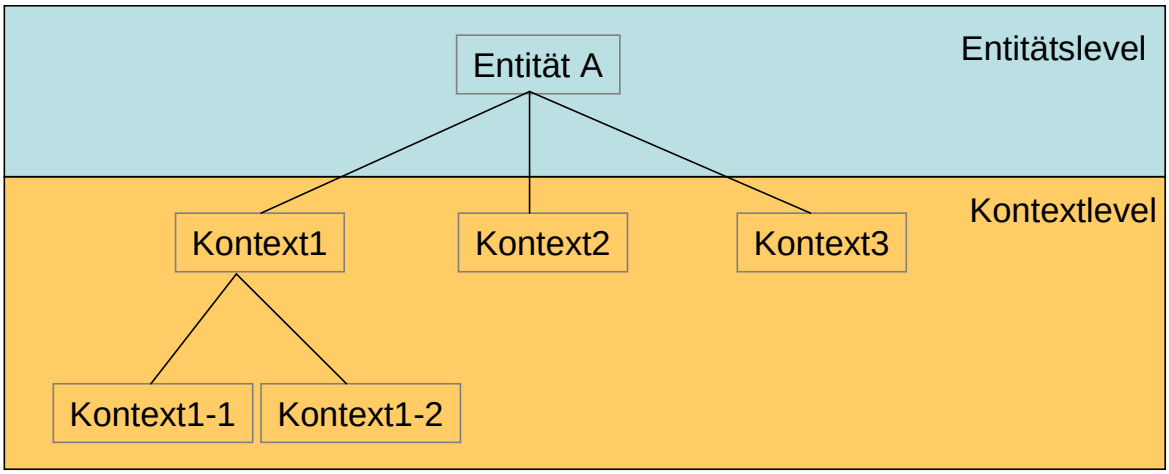


Abbildung 10: Ebenen der Identitätsprüfung

3. Rollen von Objekten in der objekt-orientierten Modellierung

Nach einer Einführung ins Semantic Web soll nun eine Einführung in den zweiten Themenschwerpunkt dieser Arbeit gegeben werden, nämlich in Rollen als Konstrukte der konzeptuellen objekt-orientierten Modellierung. Nach einer Darstellung der Sichten auf Rollen wird auf zwei ähnliche Rollenmodelle eingegangen, welche zu den meist zitierten Rollenmodellen in der objekt-orientierten Welt gehören. Diese Rollenmodelle dienen im Teil B dieser Arbeit als Basis für die Erstellung eines Rollenmodells für das Web of Data.

3.1 Einleitung und Überblick

Wenn man von Rollenmodellen spricht, stellt sich zuallererst die Frage was eine Rolle ist. Das renommierte Merriam-Webster's Online Dictionary [Mer09] liefert folgende Bedeutungen für das Wort „role":

- a character assigned or assumed <had to take on the *role* of both father and mother>
- a socially expected behaviour pattern usually determined by an individual's status in a particular society
- a part played by an actor or singer
- a function or part performed especially in a particular operation or process <played a major *role* in the negotiations>

Demnach ist eine Rolle ein Konzept (bzw. Klasse), dessen Instanzen in „some particular pattern of relationships" involviert sind [Gua92].

Für einen Überblick über bestehende Ansätze zur Modellierung und Verwendung von Rollen sei auf [ZhZh08] und [Ste00] verwiesen. Rollen sind eines der umstrittensten Konstrukte in konzeptueller Modellierung [Oli07]. Auch [Ste00] ist ähnlicher Ansicht: „ … although there appears to be a general awareness that roles are an important modelling concept, until now no consensus has been reached as to how roles should be represented or integrated into the established modelling frameworks. This may partly be due to the different contexts in which roles are introduced, …". [Ste00] hat in etwa 25 verschiedene Rollenmodelle analysiert und auf Gemeinsamkeiten hin untersucht. Dabei ist er zum Schluss gekommen, dass trotz der vielen Ansätze die Anzahl an substanziell unterschiedlichen Definitionen klein ist. Er hat drei Gruppen von Sichten auf Rollen identifizieren:

- Rollen als benannte Stelle,
- Rollen als Spezialisierung und/oder Generalisierung und
- Rollen als Zusatzinstanzen.

3.1.1 Rollen als benannte Stelle

Die einfachste Bedeutung einer Rolle ist als benannte Stelle in einer Beziehung [Ste00]. Jeder Beteiligte in einem Beziehungstyp spielt eine Rolle in dieser [Oli07]. In UML Klassendiagrammen beispielsweise kann eine Assoziation zwischen zwei Klassen auch durch Rollennamen dargestellt werden, nicht nur durch Assoziationsnamen. Existiert z.B. eine Beziehung zwischen Person und Ort mit dem Beziehungsnamen „wohnt“, so kann diese auch ausgedrückt werden, indem dem Assoziationsende bei der Klasse Person der Rollenname „Bewohner“ und dem Assoziationsende der Klasse Ort der Rollenname „Wohnort“ zugewiesen wird [SeGu06]. In dieser Bedeutung ist eine Rolle „simply a name given to a participation of an entity type“ [Oli07]. D.h. eine Rolle existiert nur im Kontext von Beziehungen [Ste00] [Gua92].

3.1.2 Rollen als Spezialisierung und/oder Generalisierung

Die Ansätze welche dieser Kategorisierung zugeordnet sind, betrachten sowohl natürliche Typen (das sind Objekttypen wie beispielsweise Person) als auch Rollen als Typen und definieren die Beziehung zwischen den beiden als eine Subklassenbeziehung bzw. Superklassenbeziehung. D.h. der Rollentyp ist spezieller als der natürliche Typ seines Rollenspielers und deshalb ist der Rollentyp eine Spezialisierung des natürlichen Typs, in manchen Ansätzen auch vice versa. Beispielsweise, wenn Person der natürliche Typ ist und Angestellter und Student die Rollentypen, dann sind Angestellter und Student Subklassen von Person. Eine Implementierung von Rollen nach diesem einfachen Ansatz bringt allerdings einige Probleme mit sich. Zum einen kann, wenn ein natürlicher Typ sowohl natürliche Typen als auch Rollentypen als Subklassen hat, nicht mehr zwischen den beiden unterschieden werden. Zum anderen gibt es Probleme wenn eine Rolle von mehreren natürlichen Typen eingenommen werden kann. Beispielsweise wenn sowohl eine Organisation als auch eine Person die Rollen Kunde und Lieferant einnehmen können. Die Lösung dieser Probleme würde in der Trennung von natürlichen Typen und Rollentypen liegen, sodass beide in einer separaten Hierarchie existieren, wie das bei Rollen als Zusatzinstanzen der Fall ist.

Rollen als Spezialisierung und/oder Generalisierung setzen eine dynamische und mehrfache Klassifizierung voraus, nachdem eine Person Rollen wechseln kann und mehrere Rollen spielen kann. Diese Rollenansätze bringen auch mit sich, dass ein Objekt und dieses Objekt in seinen Rollen als eine einzige Instanz repräsentiert wird. Diese Instanz hat nur eine Beschaffenheit und eine Identität.

3.1.3 Rollen als Zusatzinstanzen

Diese Gruppe von Rollenmodellen charakterisiert, dass Rollen unabhängige Typen haben, deren Instanzen Träger von rollenspezifischen Zuständen und Verhalten sind, in den meisten Rollenmodellen jedoch nicht Träger der Identität sind. Das Objekt, welches Träger der Identität ist, und seine Rollen stehen in einer played-by Beziehung. D.h. ein role player (Objekt) kann verschiedene Rollen einnehmen und dabei formen das Objekt und seine Rollen ein Aggregat. Rollen können dynamisch angenommen werden, was zu einer Neuerstellung einer Instanz eines Rollentyps führt, welche in das Aggregat aufgenommen wird. Das Verlassen einer Rolle führt dazu, dass die Rolle vom player getrennt und gelöscht wird.
Diese Klasse beinhaltet die meisten Rollenmodelle und kann als die einzig legitime objekt-orientierte Umsetzung von Rollen betrachtet werden [Ste00]. Diese Klasse beinhaltet unter anderem zwei Ansätze, welche in [WJS95] und [GSR96] vorgestellt werden. Auf diese wird nun genauer eingegangen.

3.2 Ausgewählte Rollenmodelle aus der objekt-orientierten Modellierung

In der objekt-orientierten Welt sind Rollen als intuitives und wichtiges Konstrukt zum Modellieren von Phänomenen aus der realen Welt erkannt worden [Kri95] [ZhZh08]. [GSR96] begründen die Notwendigkeit eines Rollenmodells für die objekt-orientierte Modellierung mit der sich im Laufe der Zeit entwickelnden Entitäten. Eine Person etwa kann im Laufe der Zeit verschiedene Rollen annehmen und in Rollen, wie beispielsweise Student, Angestellter oder Projektmanager, schlüpfen. Solche sich entwickelnden Objekte mittels Klassenhierarchien darzustellen, bei denen Objekte nur zu einer einzigen Klasse gehören können, ist eine sehr aufwändige und mühsame Angelegenheit Denn Objekte müssen neu klassifiziert werden sobald sie sich weiterentwickeln. Die Angelegenheit verkompliziert sich ungemein, sobald eine Entität mehrere Rollen annimmt oder gar eine Rolle mehrmals annimmt. [GSR96]

Im Folgenden wird nun auf die zwei der meist zitierten Rollenmodelle aus der objekt-orientierten Welt eingegangen, nämlich auf das Rollenmodell nach [GSR96] und nach [WJS95].

3.2.1 Das Rollenmodell nach [GSR96]

Im Ansatz von [GSR96] wird ein Objekt aus der realen Welt durch mehrere Objekte repräsentiert. Ein Objekt repräsentiert das Objekt aus der realen Welt in einer bestimmten Rolle. Die Objekte einer Entität werden in einer Objekthierarchie organisiert, in der spezielle Objekte von generellen Objekten variable Werte und Methoden erben. Ist z.B. jemand Student und Angestellter, dann wird er durch ein Person-Objekt, ein Student-Objekt und ein Angestellter-Objekt repräsentiert, wobei das Student-Objekt und das Angestellter-Objekt die Attribute vom Person-Objekt erben. D.h. es findet eine Spezialisierung auf Instanzebene statt und Objekte erben voneinander auf Instanzebene. Der Grundgedanke dieses Ansatzes ist, dass es eine Rollenhierarchie gibt, welche ein Baum von Rollentypen ist, wobei die Wurzel dieses Baumes ein Objekttyp ist, der zeitunveränderliche Eigenschaften enthält. Alle anderen Knoten enthalten Eigenschaften, welche das Objekt im Laufe seines Lebens annehmen oder abstoßen kann. Zu jedem Zeitpunkt ist eine Entität durch eine Instanz des Wurzeltyps und eine Instanz jedes Rollentyps, welche es gerade einnimmt, repräsentiert. Nimmt eine Entität eine neue Rolle an, so wird eine neue Instanz des Rollentyps erzeugt, verlässt es hingegen eine Rolle, so wird die rollenspezifische Instanz gelöscht.

3.2.1.1 Eigenschaften von Rollen nach [GSR96]

Im Rollenmodell von [GSR96] haben Rollen folgende Eigenschaften:

- Verschiedene Rollen einer Entität können eine gemeinsame Struktur und Verhalten teilen, beispielsweise die Studentenrolle und die Angestelltenrolle einer Person teilen den Namen und das Geburtsdatum.
- Entitäten können Rollen dynamisch annehmen und verlassen, beispielsweise eine Person in der Rolle als Angestellte kann befördert werden zur Abteilungsleiterin und später vielleicht, wenn sie den Job nicht gut macht wieder degradiert werden.
- Rollen können unabhängig voneinander angenommen und verlassen werden, beispielsweise eine Person kann ein Student werden, unabhängig von der Rolle als Angestellter.

- Entitäten können ein rollenspezifisches Verhalten aufweisen, beispielsweise kann eine Person in ihren Rollen als Student, Angestellter und Projektmitarbeiter unterschiedliche Telefonnummern haben.
- Rollen beschränken den Zugriff zu einem speziellen Kontext, beispielsweise das Gehalt eines Angestellten ist nicht zugänglich, wenn die Person in der Studentenrolle betrachtet wird.
- Entitäten können mehrmals eine Rolle desselben Rollentyps annehmen, beispielsweise ein Angestellter kann mehrmals die Rolle als Projektmitarbeiter in verschiedenen Projekten annehmen.

3.2.1.2 Rollenhierarchien im Ansatz von [GSR96]

Eine Rollenhierarchie besteht aus einem Baum von Rollentypen und beschreibt wie sich eine Instanz eines Rollentyps weiterentwickeln kann. Auf Schemaebene sieht eine Rollenhierarchie gleich wie eine Klassenhierarchie aus. Jeder Rollentyp definiert Instanzvariablen und Instanzmethoden. Der Unterschied zwischen einer Rollenhierarchie und einer Klassenhierarchie ist, dass bei ersterer Subtypen keine Instanzvariablen und Instanzmethoden vom Supertyp erben. Die Vererbung findet stattdessen auf Instanzebene statt. In Abbildung 11 ist eine Rollenhierarchie mit Ausprägungen dargestellt. Rollentypen werden als transparente Kreise dargestellt, Rolleninstanzen als blau gefärbte Kreise. Jeder Rollentyp ist mit einem dicken, grauen Pfeil mit jenen Rollentypen bzw. Klassen verbunden, dessen Instanzen diese Rolle einnehmen dürfen. Instanzen desselben Objektes aus der realen Welt sind mit einem dünnen, schwarzen Pfeil verbunden, welcher eine Rollenbeziehung ausdrückt. Doppelt umrandete Kreise sind qualifizierte Rollen, welche später genauer erläutert werden. Auf Instanzebene ist ein Objekt aus der realen Welt durch eine Instanz des Wurzeltyps und als eine Instanz von jedem Rollentyp welchen es einnimmt dargestellt. Beispielsweise ist Peter Black als eine Instanz von Person, eine Instanz von Student, zwei Instanzen von Employee und einer Instanzen von Projekt Manager dargestellt. Tina White ist durch eine Instanz von Person dargestellt. Wie schon erwähnt werden Instanzvariablen zwischen Rollentypen nicht vererbt, somit stehen bei jeder Rolleninstanz nur jene Variablen, die rollenspezifisch sind. Es findet allerdings eine Vererbung zwischen den Instanzen statt, indem Nachrichten, die von der Subrolle nicht verstanden werden, an die Superrolle weitergeleitet werden.

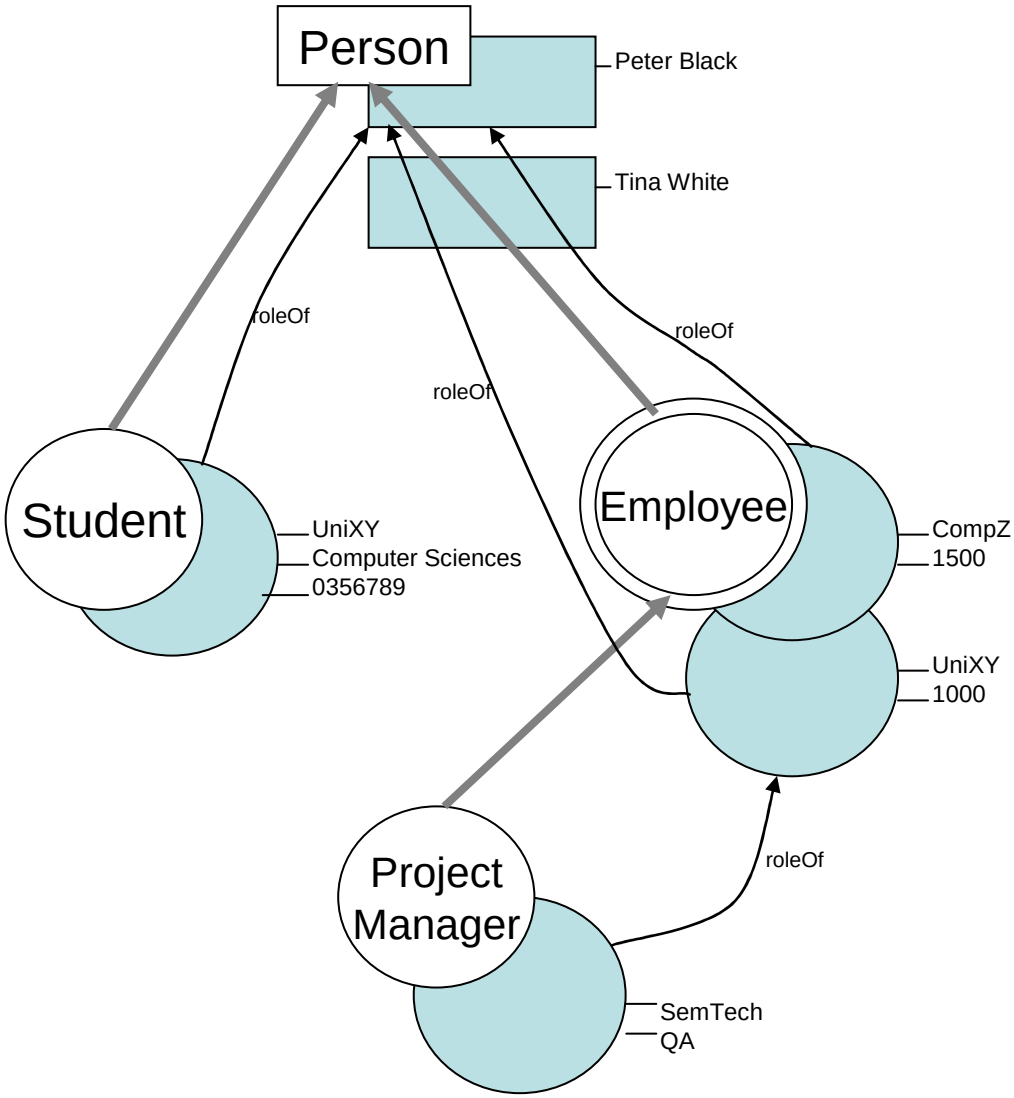


Abbildung 11: Rollenhierarchie im Rollenansatz von [GSR96]

3.2.1.3 Qualifizierte Rollen

[GSR96] definieren eine qualifizierte Rolle wie folgt:

„A role type defines a qualified role if a real-world entity may have several occurrences of that role. Occurrences of the same entity are distinguished by the value of a special instance variable named qualifier."

In Abbildung 11 definiert der Rollentyp "Employee" eine qualifizierte Rolle mit dem Arbeitgeber als qualifizierendes Attribut. Peter Black ist bei zwei verschiedenen Firmen angestellt und wird unter anderem durch zwei Instanzen vom Rollentyp „Employee" repräsentiert. In den beiden Anstellungen hat er ein anderes Einkommen, nämlich bei der UniXY verdient er „1000" und der CompZ verdient er „1500".

3.2.1.4 Kombination von Klassen- und Rollenhierarchien

Im Ansatz von [GSR96] wird ein einzigartiges Feature vorgestellt, welches es ermöglicht die Rollenhierarchie mit der Klassenhierarchie zu kombinieren. Eine Klassenhierarchie ermöglicht die Klassifikation von Entitäten in eine Menge von disjunkten Entitätstypen. Die Rollenhierarchie hingegen definiert unterschiedliche Kontexte für Entitäten desselben Typs.

Eine Klassenhierarchie kann mit mehreren Rollenhierarchien so zusammengefügt werden, dass jede Klasse in der Klassenhierarchie als Wurzel für eine Rollenhierarchie und jeder Rollentyp als Wurzel für eine Klassenhierarchie fungiert, d.h. Rollen können spezialisiert werden. Rollenhierarchien werden entlang der Klassenhierarchie vererbt.

Abbildung 12 zeigt eine Klassenhierarchie und zwei Rollenhierarchien, wobei auf die Angabe von Instanzvariablen und Instanzmethoden aus Vereinfachungsgründen verzichtet worden ist. Die Klassenhierarchie besteht aus drei Objektklassen, nämlich LegalEntity und ihren Subklassen Company und Person. Die Klassen LegalEntity und Person sind Wurzeln für Rollenhierarchien. LegalEntity ist die Wurzel für die Rollenhierarchie, bestehend aus der qualifizierten Rolle Customer. Die Objektklasse Person ist die Wurzel für die zweite Rollenhierarchie bestehend aus den Rollen Employee, ProjectManager, Student und ForeignStudent. Die Rollenhierarchie bestehend aus der qualifizierten Rolle Customer wird von den Subklassen von LegalEntity geerbt. D.h. dass jede Instanz der Objektklassen Person und Company, sowie jede Rolle, die in der Rollenhierarchie von Person definiert ist, die Rolle Customer annehmen kann.

Wie schon erwähnt können Rollen nicht nur Subrollen, sondern auch Subklassen haben. Ersteres ist der Fall zwischen Employee und ProjectManager, wobei ProjectManager eine Subrolle von Employee ist. Letzteres ist der Fall zwischen Student und ForeignStudent. ForeignStudent ist eine Spezialisierung von Student und es werden alle Instanzvariablen und Instanzmethoden von Student zu ForeignStudent vererbt. Eine derartige Spezialisierung bedeutet, dass alternativ zu einer Studentenrolle eine Rolle als ForeignStudent mit weiteren Merkmalen angenommen werden kann.

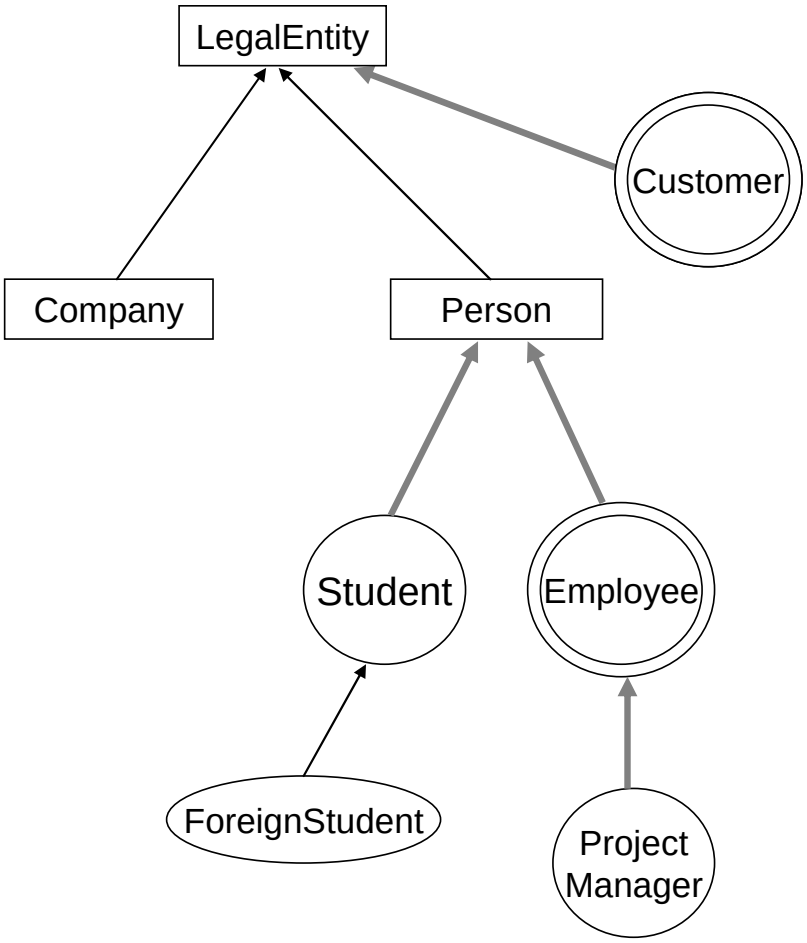


Abbildung 12: Kombination von Rollen- und Klassenhierarchie im Ansatz von [GSR96]

3.2.2 Das Rollenmodell nach [WJS95]

[WJS95] haben unabhängig von [GSR96] ein sehr ähnliches Rollenmodell entwickelt, wobei [WJS95] den Fokus auf die Objektmigration gelegt hat. Demnach ergänzen sie das Rollenmodell von [GSR96] um eine Unterscheidung bei den Objektklassen. Sie führen neben Rollenklassen auch noch dynamische und statische Objektklassen ein. Eine Instanz einer statischen Objektklasse kann nicht migrieren zu einer anderen Objektklasse, während eine Instanz einer dynamischen Objektklasse dies schon kann. Beispielsweise können Instanzen der Klasse car, welche eine statische Subklasse von vehicle ist, nicht zu anderen Klassen, beispielsweise zu motorcycle oder bicycle, welche auch statische Subklassen von vehicle sind, migrieren. Andererseits, wenn Employee eine dynamische Subklasse von Person ist, so können Instanzen von Person, welche nicht vom Objekttyp Employee sind, zu dieser Objektklasse migrieren. Die Unterscheidung zwischen den drei Klassen (Rollenklasse, statische und dynamische Objektklasse) ist orthogonal, sodass

beispielsweise dynamische Subklassen von Objekt- oder Rollenklassen oder auch Rollenklassen von dynamischen und statischen Objektklassen existieren können.

Eine Rollenklasse unterscheidet sich von einer Subklasse dadurch, dass es zu einer Instanz einer Superklasse mehrere Instanzen der Rollenklasse geben kann. Eine Instanz einer Subklasse ist identisch (haben denselben Identifier) mit einer Instanz ihrer Superklasse, während eine Instanz einer Rollenklasse niemals identisch ist mit Instanzen ihrer Player-Klasse. Dabei verweisen [WJS95] auf das Counting-Problem, auf welches zu einem späteren Zeitpunkt noch einmal genauer eingegangen wird. Beispielsweise, wenn Employee eine Subklasse von Person ist, dann ist die Anzahl an Employee-Instanzen gleich der Anzahl an Person-Instanzen, gezählt in einer Menge von Employee-Instanzen. Anders, wenn Employee eine Rolle von Person ist, dann kann sich die Anzahl an Instanzen von Employee von der Anzahl an Instanzen von Person unterscheiden, da bei Rollenklassen die Möglichkeit besteht, dass sie mehrfach von ihrem Player eingenommen werden können.

Teil B – Rollen im Web of Data

4. Rollen im Web of Data – Motivation, Analyse, Anforderungen 54
5. Rollen im Web of Data – Umsetzung mit RDF(S) 85

4. Rollen im Web of Data – Motivation, Analyse, Anforderungen

Im Folgenden wird ein Rollenmodell für das Web of Data entwickelt. Zuerst wird die Motivation für ein Rollenmodell erläutert und erklärt welchen Nutzen ein Rollenmodell für das Web of Data hat. Anschließend werden vier Anwendungsfälle vorgestellt, anhand deren die Anforderungen an ein Rollenmodell im Web of Data abgeleitet werden. Aufgrund der Anforderungen wird ein Ansatz aus der objekt-orientierten Welt untersucht und die Unterschiede dessen Rahmenbedingungen zu den Rahmenbedingungen im Web of Data herausgearbeitet. Folge dessen wird ein Rollenmodell für das Web of Data vorgestellt und anschließend analysiert.

Wir verwenden die Begriffe Rolle, Rollenklasse und Rollentyp folgendermaßen:
Eine Rolle (z.B. Mr. Black als Employee bei UniXY) ist eine kontextspezifische Beschreibung einer Nicht-Informationsressource (z.B. Mr. Black). Rollen werden zu Rollenklassen (z.B. EmployeeRole) zusammengefasst. Die Begriffe Rollentyp und Rollenklasse werden im Weiteren synonym verwendet.

4.1 Motivation für Rollen im Web of Data

In diesem Abschnitt wird erläutert, warum Rollen im Web of Data einen wesentlichen Mehrwert bringen. Die Motivation für den Einsatz von Rollen im Web of Data stützt sich auf drei Grundargumenten:

a) Ebenso wie Rollen für die objekt-orientierten Welt ein Konstrukt sind um Phänomene aus der Realität intuitiver darzustellen, sind sie das auch für das Web of Data. Sie stellen eine wesentliche Erweiterung der Ausdrucksstärke dar.

Darüber hinaus gilt im Web of Data:

b) Das Rollenmodell ist ein wesentlicher Beitrag zur Lösung des Coreferenz-Problems im Web of Data.
c) Das Rollenmodell fördert die Navigierbarkeit im Web of Data.

4.1.1 Argument a): Wesentliche Erweiterung der Ausdrucksstärke

Wie bereits in Kapitel 3 erläutert, sind Rollen eine kontextspezifische Beschreibung einer Entität, wobei eine Entität mehrere Rollen einnehmen kann und es somit mehrere kontextspezifische Beschreibungen von ein und derselben Entität geben kann. Es ist sogar möglich, dass eine Entität eine Rolle mehrfach einnimmt, d.h. es gibt mehrere

Beschreibungen einer Entität im gleichen Kontext, beispielsweise, wenn eine Person mehrere Angestelltenverhältnisse hat und somit mehrere Angestelltenrollen einnimmt. Außerdem können so genannte Rollenhierarchien (vgl. Kapitel 4.4.2) entstehen, d.h. eine Rolle kann wiederum andere Rollen einnehmen, beispielsweise kann eine Angestelltenrolle eine Rolle als Projektmanager einnehmen.
Diese Rollensemantik, d.h. eine kontextspezifische Beschreibung von Entitäten und eine hierarchische Anordnung dieser, bilden eine wichtige und wesentliche Erweiterung der Ausdruckmöglichkeiten im Web of Data, sodass die reale Welt direkter und genauer abgebildet werden kann.

Um der Behauptung gerecht zu werden, dass das Rollenmodell eine Erweiterung im Web of Data ist, soll gezeigt werden, dass diese Semantik im Web of Data, mit dem zur Verfügung stehenden Standardvokabular nicht nachgebildet werden kann.

Im Web of Data entsprechen Entitäten Nicht-Informationsressourcen und alle beschreibungsrelevanten Eigenschaften der Entität werden mittels Aussagen über die Nicht-Informationsressource getätigt. [Boo06] erklärt, dass durch die Beschreibung einer Nicht-Informationsressource eine Art Modell einer Entität entsteht (vgl. Kapitel 2.5.2.1), das als Stellvertreter für die Ressource manipuliert werden kann. D. h., wenn Entitäten mehrfach (kontextspezifisch) beschrieben werden sollen, muss ein Objekt aus der realen Welt durch mehrere unabhängige Modelle abgebildet werden. Die Rollensemantik angewandt auf das Web of Data wird in Abbildung 13 verdeutlicht. Die verschiedenen Modelle von Mr. Black stehen in einer Rollebeziehung zueinander. Dies wird durch die grauen, dicken Pfeile dargestellt. Auf dieser Basis soll nun geprüft werden, ob Rollen mit dem Standardvokabular im Semantic Web abbildbar sind, d.h. eine Semantik nachbildbar ist, die ausdrückt, dass zwischen verschiedenen Beschreibungen (Modellen) einer Nicht-Informationsressource getrennt werden kann und diese gleichzeitig Beschreibungen über ein und dieselbe Nicht-Informationsressource sind.

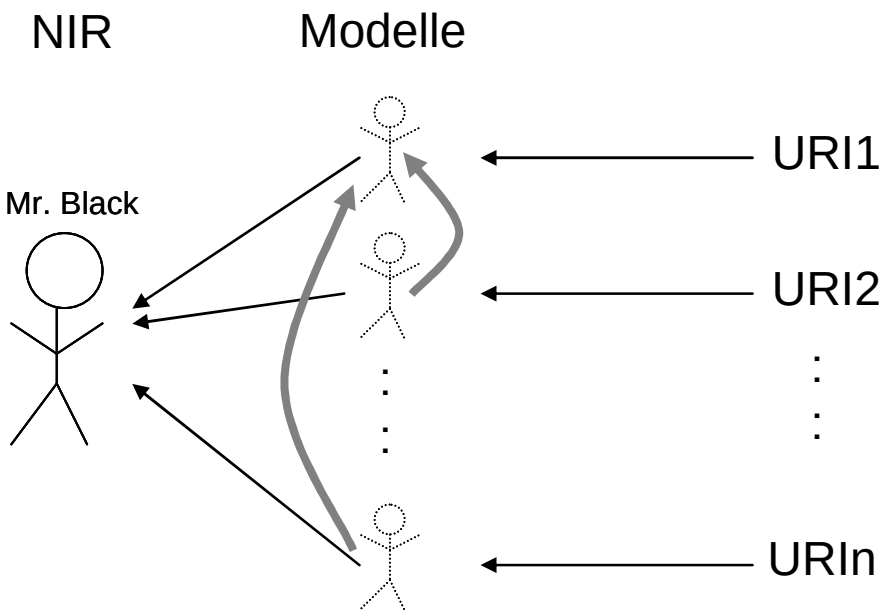


Abbildung 13: Rollenmodell im Web of Data

Eine Verlinkung der Beschreibungen mittels owl:sameAs hat zur Folge dass aus mehreren unabhängigen Beschreibungen einer Nicht-Informationsressource eine vereinigte Beschreibung dieser entsteht und führt zu einem anderen Ergebnis (siehe Abbildung 8 in Kapitel 2.6.1), welches nicht der Rollensemantik entspricht. Während eine Verlinkung mittels owl:sameAs zu stark ist, erfüllen andere Möglichkeiten, wie beispielsweise rdfs:seeAlso nicht den Zweck. Ein Verweis mittels rdfs:seeAlso wird verwendet um auszudrücken, dass in einer anderen Informationsressource (Objekt-URI) Informationen zur Subjekt-URI gefunden werden können [MaMi04b].

Eine mögliche Repräsentation von Rollen im Web of Data könnte mittels einer n-ären Beziehung gemacht werden [NoRe04]. Nachdem eine Rolle als eine kontextspezifische Beschreibung einer Entität definiert ist, könnte man argumentieren nicht die Entität zum Interessensobjekt und in den Mittelpunkt der Beschreibung zu stellen, sondern die Beziehung in deren Kontext die Rolle existiert. Beispielsweise, um eine Person in der Rolle als Angestellten zu beschreiben, könnte das Interessensobjekt die Anstellung sein (siehe Abbildung 14). An die Anstellung, als zentrales Beschreibungsobjekt, werden alle rollenspezifischen Attribute gehängt. Einerseits ist dies aus meiner Sicht keine realitätsnahe Abbildung, da die Attribute die Person beschreiben und nicht die Anstellung, andererseits ist dies notwendig um eine rollenspezifische Trennung von Attributen zu erreichen. Weiters müßte für jeden Rollentyp jedes Prädikat neu definiert werden, weil ansonsten Domainangaben von Prädikaten zu Falschklassifizierungen führen würden (vgl.

Kapitel 2.5.4.1), dadurch wäre die Wiederverwendbarkeit von Ontologien stark eingeschränkt. Beispielsweise das Prädikat phone, welches die Telefonnummer in der jeweiligen Rolle definiert muss jeweils für die Anstellung und für die Studentenstelle definiert werden, da ansonsten die zentralen Beschreibungsobjekte gleichermaßen klassifiziert werden würden.

```
x:employment123 a x:Employment;
        x:employee y:black;
        x:employer z:compZ;
        x:salary „1500“.

x:employment124 a x:Employment;
        x:employee y:black;
        x:employer u:uniXY;
        x:salary „1000“.
```

Abbildung 14: Rollen als n-äre Beziehung am Beispiel von Mr. Black in den Rollen als Employee

Weiters ist dieser Ansatz nicht adäquat, da durch ihn die Navigationspfade bei bereits bestehenden Ressourcen geändert werden müssen. Existiert beispielsweise eine Aussage die besagt, dass Mr. Black an der UniXY studiert und es soll eine Rolle Student erstellt werden, so muss nach diesem Ansatz die Verknüpfung zwischen Mr. Black und UniXY gelöscht werden und an das Rollenobjekt „Student“ gehängt werden. Es besteht auch die Möglichkeit, dass die Verlinkung zwischen Mr. Black und der UniXY aufrecht erhalten bleibt und die Information redundant vorhanden ist. Dann ist es allerdings keine rollenspezifische Information mehr und wenn eine Person mehrere Rollen einnimmt kommt es dadurch zu Problemen.

Zusammenfassend kann man sagen, dass es im Web of Data noch keine Möglichkeit gibt zwei unterschiedliche Modelle einer Entität zu erstellen und gleichzeitig auszudrücken, dass beide Modelle auf dieselbe Entität verweisen. Ein Modell in einem bestimmten Kontext kann auch als eine Rolle betrachtet werden. Demnach ist es im Web of Data nicht möglich Rollen wie eingangs kurz skizziert darzustellen. Wünschenswert wäre also eine Darstellung wie in Abbildung 13, wo zwei URIs verschiedene Modelle von Mr. Black (als Stellvertreter von Mr. Black), welche in einer Rollenbeziehung zueinander stehen (dargestellt durch den dicken grauen Pfeil), benennen und gleichzeitig ausgedrückt wird,

dass es Modelle derselben Nicht-Informationsressource, nämlich von Mr. Black, sind. Dies entspricht weitestgehend dem, was durch ein Rollenmodell ausgedrückt wird. Durch ein Rollenmodell kann sogar noch mehr ausgedrückt werden, worauf jedoch zu einem späteren Zeitpunkt detailliert eingegangen wird.

4.1.2 Argument b): Beitrag zur Lösung des Coreferenz-Problems

Durch die Semantik von Rollen liefert das Rollenmodell einen Beitrag zur Lösung des Problems, wenn verschiedene URIs dieselbe Entität benennen und diese voneinander unabhängig sind. Dieses Problem wird von [JGM07] und [JGM08] als das Coreferenz-Problem betitelt (vgl. Kapitel 2.6.2).

Ein Rollenmodell liefert insofern einen Beitrag zur Lösung des Coreferenz-Problems, als dass es einerseits kontextspezifische Beschreibungen einer Entität zulässt und andererseits diese Beschreibungen identitätsspezifisch bündelt. D.h. das Rollenmodell bietet die Möglichkeit, dass eine Entität mehrfach kontextspezifisch beschrieben wird und gleichzeitig ausgedrückt wird, dass sich diese Beschreibungen (Rollen) auf dieselbe Entität beziehen (vgl. Abbildung 13). Wobei letzteres einen Lösungsbeitrag für das Coreferenz-Problem darstellt. Denn, befinden sich zwei URIs in einer Rollenbeziehung zueinander, so drückt das unter anderem aus, dass sie dieselbe Entität identifizieren (vgl. Abbildung 13).

4.1.3 Argument c): Förderung der Navigierbarkeit

Im Web of Data können Nicht-Informationsressourcen nicht nur autoritativ beschrieben werden, sondern auch nicht-autoritativ. Letzteres schränkt jedoch die Navigierbarkeit stark ein, weil die nicht-autoritative Beschreibung nicht direkt referenziert werden kann (vgl. Kapitel 2.5.2.2).

Das Rollenmodell fördert autoritative Beschreibungen und somit die Navigierbarkeit. Es bietet die Möglichkeit einer verteilten autoritativen Beschreibung einer Nicht-Informationsressource, d.h., dass eine Nicht-Informationsressource mehrfach in unterschiedlichen Informationsressourcen autoritativ beschrieben wird. Durch die Rollensemantik kann allerdings gleichzeitig ausgedrückt werden, dass es sich um dieselbe Nicht-Informationsressource handelt. Nicht-autoritative Beschreibungen einer Nicht-Informationsressource können also durch den Einsatz des Rollenmodells ersetzt werden, zugunsten einer besseren Navigierbarkeit.

4.2 Anforderungen an Rollen im Web of Data

Nachdem gezeigt worden ist, dass ein Rollenmodell eine große Bereicherung für das Web of Data ist, sollen nun die Anforderungen an ein solches definiert werden. Dabei wird zwischen semantischen und situationsbedingten Anforderungen unterschieden. Die semantischen Anforderungen sind von den situationsbedingten Anforderungen unabhängig und definieren die benötigte Ausdrucksmächtigkeit des Rollenmodells.

4.2.1 Semantische Anforderungen

Um die semantischen Anforderungen an ein Rollenmodell im Web of Data zu definieren, wird das in Kapitel 1.2 beschriebene Beispiel herangezogen.
Aus diesem Beispiel geht hervor, dass Mr. Black verschiedene Rollen einfach einnimmt und eine Rolle sogar doppelt. Die Rolle als Student, als Privatperson und als Projektmanager werden nur einfach eingenommen, während die Rolle als Angestellter doppelt eingenommen wird, einmal bei der UniXY und einmal bei der CompZ. Die Angestelltenrollen unterscheiden sich nicht im Rollentyp, trotzdem soll es möglich sein einen unterschiedlichen Gültigkeitsbereich zu definieren, welcher in diesem Fall abhängig vom Arbeitgeber ist. Diese Art von Rollen bezeichnen [GSR96] als qualifizierte Rollen (vgl. Kapitel 3.2.1.3).

4.2.2 Situationsbedingte Anforderungen

Um die situationsbedingten Anforderungen zu definieren, wird von den nun folgenden vier idealtypischen Anwendungsfällen ausgegangen, welche sich ebenfalls auf das Beispiel aus Kapitel 1.2 beziehen. Die Anwendungsfälle unterscheiden sich bezüglich der Chronologie des Erstellungszeitpunktes von Beschreibungen einer Nicht-Informationsressource und bezüglich der Aufteilung der Statements in den Informationsressourcen. Mit der Aufteilung der Statements in Informationsressourcen wird der Tatsache Rechnung getragen, dass verschiedene Autoren Kontrolle über unterschiedliche Namensräume haben.

4.2.2.1 Neuerstellung einer Rollenbeschreibung

Der erste Anwendungsfall stellt ein Szenario dar, in dem kontextspezifische Beschreibungen (Rollen) einer Nicht-Informationsressource erstellt werden sollen und es noch keine Beschreibung dieser Ressource gibt. Diese Beschreibungen werden von einem einzigen Autor erstellt. D.h. es wird eine Neubeschreibung (in diversen Kontexten) einer Nicht-Informationsressource erstellt, welche im Sinne der Navigierbarkeit und

Dereferenzierbarkeit (vgl. Kapitel 2.5.2.2) autoritative Beschreibungen sind und in einer Informationsressource getätigt werden.

Angewandt auf das durchgängige Beispiel würde das in etwa heißen, dass Mr. Black beschließt Daten über sich selbst im Web of Data zu veröffentlichen und es gibt noch keine Beschreibung von ihm im Web of Data. Mr. Black möchte sich dabei in den diversen Kontexten selbst beschreiben, d.h. er gibt sich selbst URIs (eine pro Rolle), die in einem Namensraum liegen, über den er Kontrolle hat. Die Beschreibungen fasst er in ein Dokument zusammen.

Abbildung 15 stellt diese Situation vereinfacht dar. URI 1 identifiziert Mr. Black in einem allgemeinen Kontext. URI 2 identifiziert Mr. Black in einem spezielleren Kontext als URI 1. Der dicke graue Pfeil drückt die Rollenbeziehung zwischen den beiden aus. Im Sinne der Navigierbarkeit und Dereferenzierbarkeit (vgl. Kapitel 2.5.2.2) werden autoritative Beschreibungen erstellt und diese in einer Informationsressource zusammengefasst.

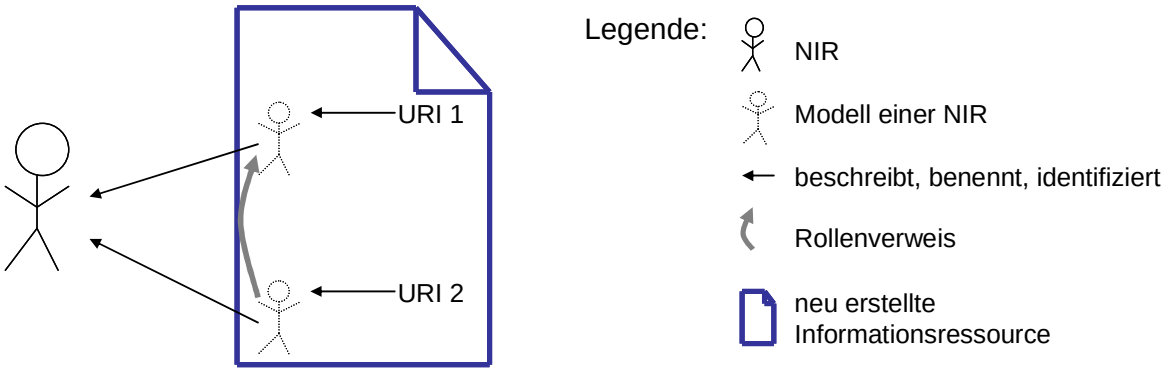


Abbildung 15: Neuerstellung einer Rollenbeschreibung

4.2.2.2 Erstellung einer Spezialisierungsrolle

Dieser Anwendungsfall stellt ein Szenario dar, in dem es bereits eine allgemeine Beschreibung einer Nicht-Informationsressource gibt und es soll eine Beschreibung in einem spezielleren Kontext, als der Kontext der bereits existierenden Beschreibung, erstellt werden. Die Beschreibung im allgemeinen Kontext der Entität wird zur Superrolle für die Beschreibung im spezielleren Kontext der Entität. D.h. es wird eine Rollenbeziehung zu einer bereits existierenden allgemeinen Beschreibung derselben Nicht-Informationsressource erstellt. Weiters wird die Beschreibung im speziellen Kontext in einem anderen Namensraum gemacht, als in jenem der allgemeinen Beschreibung, da sich

die Autoren unterscheiden und diese die Kontrolle über verschiedene Namensräume haben. Im Sinne der Dereferenzierbarkeit und Navigierbarkeit (vgl. Kapitel 2.5.2.2) sollen autoritative Beschreibungen erstellt werden.

Im durchgängigen Beispiel von Mr. Black ist dies beispielsweise der Fall, wenn Mr. Black bereits von sich selbst eine allgemeine Beschreibung erstellt hat und nun möchte die CompZ, bei der er arbeitet, eine speziellere Beschreibung, nämlich im Kontext als Angestellter, erstellen und dabei auf die bereits existierende Beschreibung verweisen. Die CompZ möchte dabei die Kontrolle über die Beschreibung haben und erstellt sie deshalb im eigenen Namensraum und somit in einer anderen Informationsressource.

In Abbildung 16 ist dieser Sachverhalt vereinfacht dargestellt. URI 1 identifiziert Mr. Black in einem allgemeinen Kontext. URI 2 identifiziert Mr. Black im Kontext als Angestellten. Die Beschreibung von Mr. Black mit URI 1 existiert bereits bevor eine Beschreibung von Mr. Black mit URI 2 erstellt wird. Die Beschreibung mit URI 2, also die Rolle als Angestellter, soll neu erstellt werden. Dadurch, dass der Autor der Angestelltenrolle keine Kontrolle über den Namensraum von URI 1 hat, aber im Sinne der Navigierbarkeit und Dereferenzierbarkeit (vgl. Kapitel 2.5.2.2) eine autoritative Beschreibung erstellen möchte, erfolgt die Beschreibung mit URI 1 in einer anderen Informationsressource als die Beschreibung mit URI 2. Durch die Reihenfolge der Erstellung ergibt sich in diesem Fall, dass das Rollenstatement, welches die Rollenbeziehung ausdrückt, in der Informationsressource steht, in der URI 2 beschrieben wird.

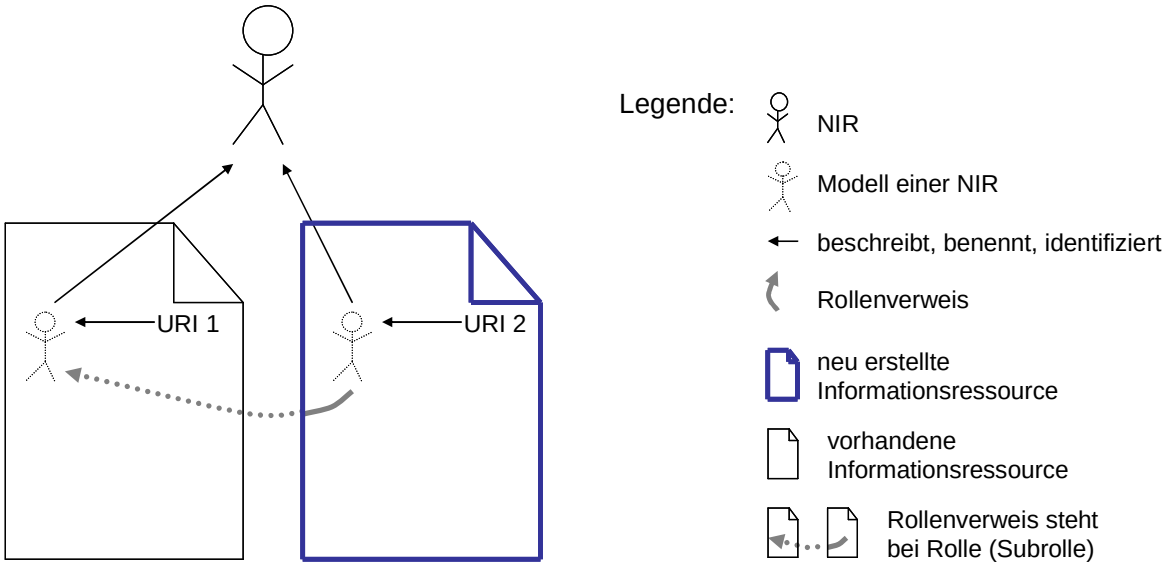


Abbildung 16: Erstellung einer Spezialisierungsrolle

4.2.2.3 Erstellung einer Generalisierungsrolle

Der dritte Anwendungsfall beschreibt ein Szenario, in dem von einer bereits existierenden Beschreibung einer Nicht-Informationsressource eine allgemeinere Beschreibung derselben Nicht-Informationsressource, als die bereits existierende Beschreibung erstellt werden soll. Dieser Anwendungsfall geht von zwei unterschiedlichen Motivationen aus, eine Generalisierungsrolle zu erstellen.

Der erste Fall (in Abbildung 17) ist das Gegenstück zum zweiten Anwendungsfall. Er beschreibt ein Szenario, in dem es bereits eine spezielle Beschreibung eines Objektes aus der realen Welt gibt. Die Beschreibung dieses Objektes soll um eine allgemeinere Rolle als die bereits existierende Rolle ergänzt werden. Dabei sollen selektive Attribute von den speziellen Rollen zur allgemeinen Rolle aufwärts vererbt werden. Eine derartige Aufwärtsvererbung soll nicht nur auf der Instanzebene, sondern auch auf der Schemaebene beschrieben werden können. Weiters wird die Beschreibung im allgemeinen Kontext in einem anderen Namensraum gemacht, als in jenem der spezielleren Beschreibung, da sich die Autoren unterscheiden und diese die Kontrolle über verschiedene Namensräume haben. Im Sinne der Dereferenzierbarkeit und Navigierbarkeit (vgl. Kapitel 2.5.2.2) sollen autoritative Beschreibungen erstellt werden.

Am durchgängigen Beispiel (vgl. Kapitel 1.2) des Mr. Black wäre das, wenn beispielsweise die CompZ bereits eine Angestelltenrolle von Mr. Black erstellt hat und

diese das allgemeine Attribut „name“ enthält. Basierend auf dieser Rolle will nun Mr. Black in seinem Namensraum eine allgemeinere Rolle von sich, als die Angestelltenrolle, erstellen und das Attribut „name“ mit seiner Ausprägung aufwärts vererben. (Anmerkung: Diese neu erstellte allgemeine Rolle kann auch als Superrolle für weitere Rollen, beispielsweise die Rolle als Privatperson, dienen.)

In Abbildung 17 ist dieser Sachverhalt vereinfacht dargestellt. URI 1 identifiziert Mr. Black in der Rolle als Angestellten. URI 3 identifiziert Mr. Black in einer allgemeinen Rolle, die beispielsweise PersonBaseRole heißen könnte. Die Beschreibung von Mr. Black als Angestellter existiert bereits. Dadurch, dass der Autor der allgemeinen Rolle keine Kontrolle über den Namensraum der speziellen Rolle hat, aber im Sinne der Navigierbarkeit und Dereferenzierbarkeit (vgl. Kapitel 2.5.2.2) eine autoritative Beschreibung erstellen möchte, erfolgt die Beschreibung mit URI 3 (allgemeine Rolle) in einer anderen Informationsressource als die Beschreibung mit URI 1. Durch die Reihenfolge der Erstellung ergibt sich in diesem Fall, dass das Rollenstatement, welches die Rollenbeziehung ausdrückt, in der Informationsressource bei URI 3 steht.

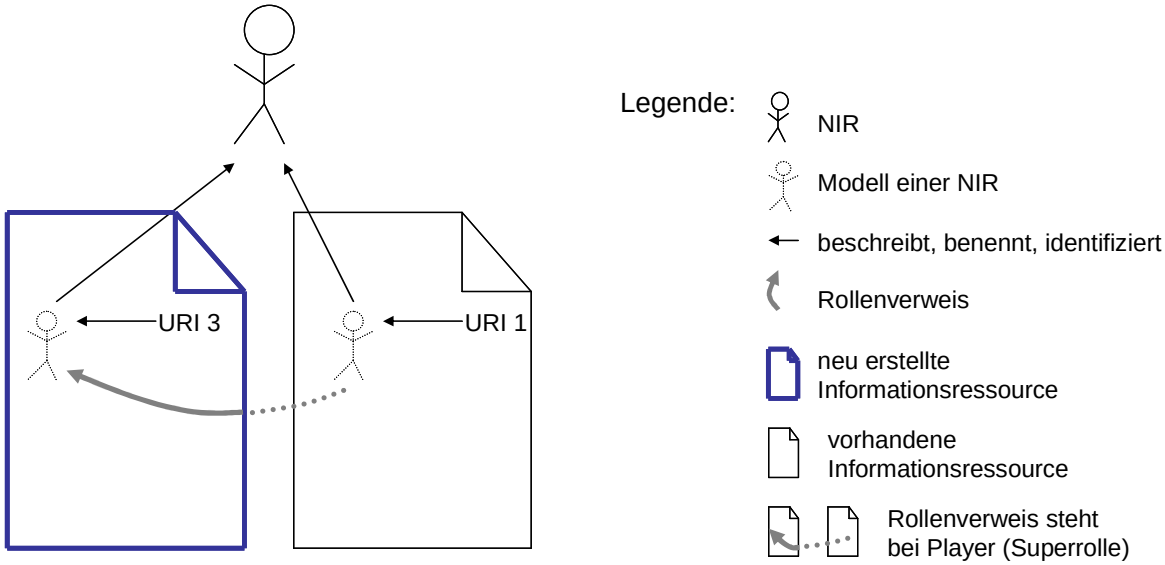


Abbildung 17: Neuerstellung einer Generalisierungsrolle

Der zweite Fall stellt eine Situation dar in der es zwei Rollen auf gleicher Ebene gibt und vorrangig ausgedrückt werden soll, dass diese zwei Beschreibungen (Rollen) von derselben Nicht-Informationsressource sind. Allerdings ist jede Rolle in einem eigenen Namensraum definiert und die Autoren haben nur die Kontrolle über den eigenen

Namensraum. Um dies auszudrücken, muss, wie im ersten Fall, eine allgemeinere Rolle als die bereits existierenden Rollen, erstellt werden. Diese allgemeine Rolle kommt jedoch ohne eine kontextspezifische Beschreibung der Nicht-Informationsressource und ohne Aufwärtsvererbung von Prädikaten aus.

Für das durchgängige Beispiel von Mr. Black würde das beispielsweise heißen, dass die CompZ eine Beschreibung von Mr. Black als Angestellter ebenso wie die UniXY eine Rolle von Mr. Black als Angestellter im Web of Data zur Verfügung stellt. Mr. Black möchte nun ausdrücken, dass beide Beschreibungen dieselbe Nicht-Informationsressource bzw. ihn identifizieren. Um dies auszudrücken, muss, da die beiden Beschreibungen auf gleicher Ebene sind, eine allgemeinere Rolle als die bereits existierenden, erstellt werden. Da Mr. Black allerdings keine Kontrolle über den Namensraum, in dem die Angestelltenrollen definiert sind, hat, muss er die Erstellung der allgemeinen Rolle und die Rollenverlinkung in einer dritten Informationsressource machen, die sich im eigenen Namensraum befindet.

In Abbildung 18 ist diese Situation vereinfacht dargestellt. Beide, URI 1 und URI 2, identifizieren Mr. Black in einer unterschiedlichen Angestelltenrolle, einmal bei CompZ und einmal bei UniXY. Dadurch, dass Mr. Black keine Kontrolle über die Namensräume, in denen die Beschreibungen existieren, hat, erstellt er im eigenen Namensraum eine neue allgemeine Rolle (identifiziert durch URI 3) mit den Rollenverlinkungen zu den Angestelltenrollen.

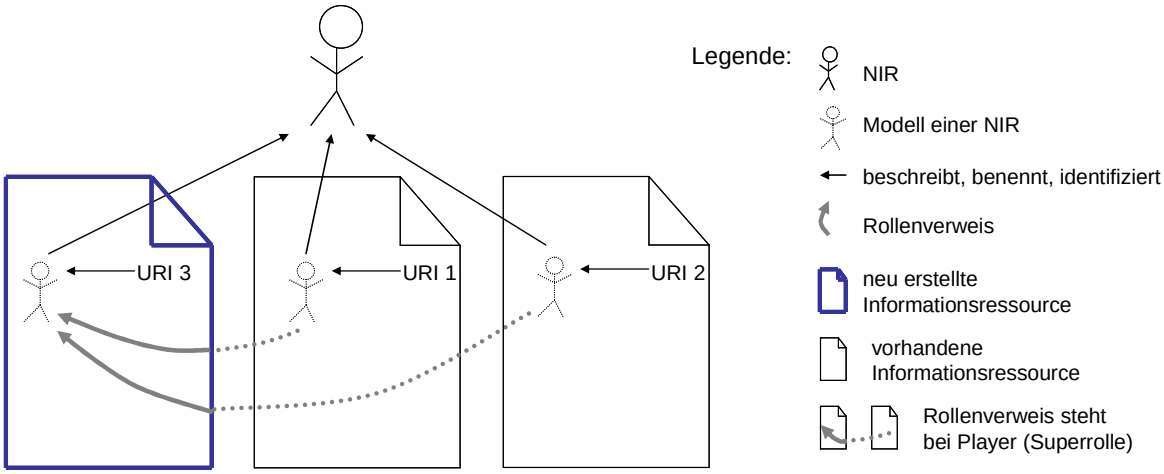


Abbildung 18: Neuerstellung einer Generalisierungsrolle

4.2.2.4 Anreicherung von vorhandenen Ressourcen

Der vierte Anwendungsfall beschreibt eine Situation, in der es bereits mehrere voneinander unabhängige Beschreibungen eines Objektes aus der realen Welt gibt. Es soll hier nicht das Interessenobjekt in einem neuen Kontext beschrieben werden, sondern die bereits existierenden Beschreibungen sollen mit dem Rollenmodell angereichert werden. D.h. die Nicht-Informationsressource ist bereits in allen gewünschten Kontexten beschrieben, jedoch sind diese Beschreibungen nicht so miteinander verlinkt, dass zum Ausdruck gebracht wird, dass sie dieselbe Ressource beschreiben. Wie schon in den vorhergehenden Anwendungsfällen handelt es sich auch hier bei den einzelnen existierenden Beschreibungen um ausschließlich autoritative Beschreibungen in unterschiedlichen Namensräumen und die einzelnen Autoren haben keine Kontrolle über den Namensraum der anderen.

Dieser Anwendungsfall würde beispielsweise für das durchgängige Beispiel von Mr. Black bedeuten, dass bereits mehrere unabhängige autoritative Beschreibungen von ihm existieren, die nicht identitätsspezifisch miteinander verlinkt sind. D.h. es gibt bereits eine allgemeine Beschreibung, eine Beschreibung als Privatperson, eine Beschreibung als Student, zwei Beschreibungen als Angestellter und eine Beschreibung als Projektmanager. Nun möchte Mr. Black zum Ausdruck bringen, dass all diese Beschreibungen ihn in verschiedenen Kontexten beschreiben bzw. die URIs mit den Beschreibungen ihn identifizieren. Um die Beschreibungen von Mr. Black anzureichern, wird eine unabhängige dritte Ressource erstellt, in der die Beziehungen zwischen den Modellen von Mr. Black, in Form von Rollenbeziehungen, definiert werden. Wird die Informationsressource mit den Rollenbeschreibungen geladen, so erhält der Benutzer dadurch einen Mehrwert.

In Abbildung 19 ist dies auszugsweise dargestellt. URI 1 und URI 2 identifizieren jeweils Mr. Black in einem unterschiedlichen Kontext. Die Beschreibungen mit URI 1 und URI 2 existieren bereits bevor von einem dritten Autor in einer dritten Informationsressource die Rollenbeziehung zum Ausdruck gebracht wird. Der Grund warum die Rollenverlinkung nicht in einer der Informationsressourcen bei einer Rolle steht, ist, dass der Autor der Rollenverlinkung keine Kontrolle über die Namensräume hat, in denen die Beschreibung der Rollen erfolgt.

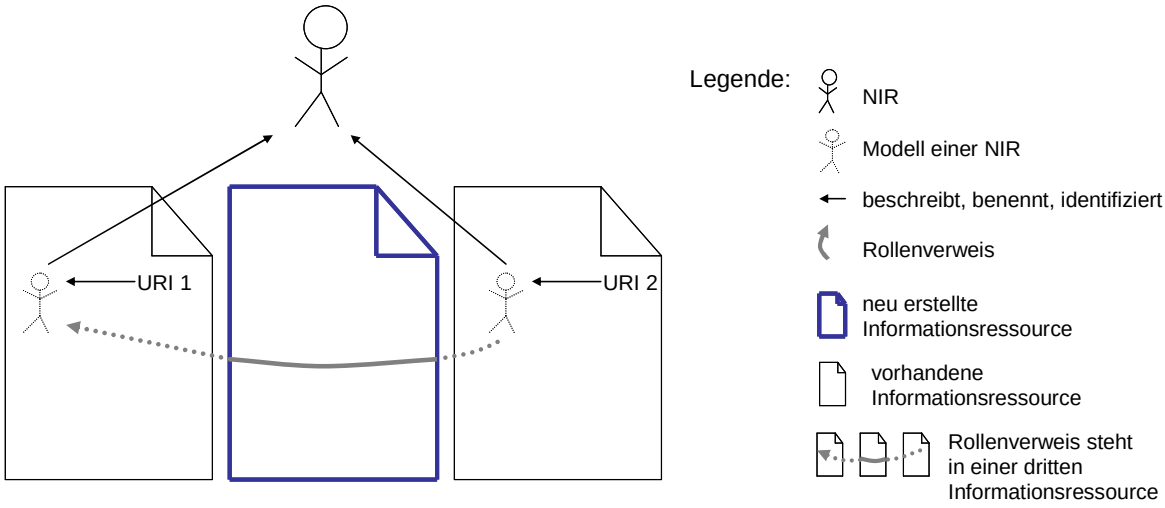


Abbildung 19: Anreicherung von vorhandenen Ressourcen

4.2.3 Zusammenfassung der Anforderungsdefinitionen

Aus den zuvor beschriebenen semantischen und situationsbedingten Anforderungsbeschreibungen werden nun die folgenden Anforderungsdefinitionen für Rollen im Web of Data abgeleitet:

Bindung der rollenspezifischen Informationen an die Entität

Dadurch, dass Rollen kontextspezifische Beschreibungen einer Nicht-Informationsressource sind, sind alle Aussagen, die in Rollen gemacht werden, für die Entität zutreffend. Demnach soll nachvollziehbar sein, welche Aussagen für die Entität Gültigkeit besitzen und welche nicht. Diese Anforderung ist beispielsweise nicht erfüllt, wenn zwei unabhängige URIs dieselbe Ressource benennen, dies jedoch nicht durch eine semantisch korrekte Verlinkung, wie beispielsweise owl:sameAs, ausgedrückt wird [DeSc04].

Rollenspezifische Trennung der Attribute

Trotz der Bindung der Attribute zur Entität soll nachvollziehbar sein, welche Aussagen für welche Rolle gültig sind. D.h. alle rollenspezifischen Attribute müssen eindeutig einer Rolle zuordenbar sein. Dies ist beispielsweise bei einer Verlinkung mittels owl:sameAs, wie in Abbildung 8 gezeigt wird, nicht erfüllt.

Kompatibilität

Dadurch, dass das Web of Data schon eine Vielzahl an Ressourcen beschreibt, ist es von außerordentlicher Wichtigkeit, dass das Rollenmodell auf bereits bestehende Daten aufgesetzt werden kann, ohne dass die Navigationspfade dieser geändert werden müssen. Bestehende Informationen sollen nur um rollenspezifische Informationen angereichert werden und diese nicht ändern. Die Kompatibilität ist beispielsweise bei den in Kapitel 4.1.1 vorgestellten Rollen als n-äre Beziehung nicht der Fall.

Wiederverwendung von Vokabularen

Im Web of Data existieren bereits eine Reihe von Vokabularen, wie z.B. das FOAF-Vokabular, um Personen und Beziehungen zwischen Personen zu beschreiben. Durch das Rollenmodell soll die Wiederverwendung solcher Vokabulare in keinster Weise eingeschränkt werden. Dies wäre beispielsweise der Fall, wenn pro Rollentyp ein separates Vokabular definiert werden müsste. Dies ist beispielsweise bei den in Kapitel 4.1.1 vorgestellten Rollen als n-äre Beziehung der Fall.

Unabhängige Klassifizierung auf Rollenebene ohne Typvererbung

Rollen sollen unabhängig von der bereits existierenden Klassifizierung, d.h. auf einer eigenen Ebene, klassifiziert werden können (vgl. Kapitel 4.4.1). Dies ist wegen dem Counting-Problem, wonach eine Mehrfachklassifizierung zu Problemen führen kann notwendig (vgl. Kapitel 4.4.1). Die Mehrfachklassifizierung kann als Nebeneffekt durch die erlaubte Verwendung von Prädikaten zur Beschreibung von Nicht-Informationsressourcen unterschiedlichen Typs entstehen, wenn die Prädikate unterschiedliche Domaineigenschaften haben. Weiters soll die Klassifizierung auf Rollenebene, im Gegensatz zur Klassifizierung auf Entitätsebene, nicht transitiv für Subrollen gelten (vgl. Kapitel 4.4.1).

Unterstützung von simplen Rollen

Simple Rollen besagen, dass eine Entität Rollen in einem nicht genau spezifizierten Kontext besitzt. Die Rollen können klassifiziert sein, beispielsweise als Student oder Angestellter, sollte jedoch eine Entität mehrere Rollen gleicher Klassifizierung annehmen, so kann über den genauen Kontext, in dem die Rolle beschrieben ist, keine Aussage gemacht werden.

Unterstützung von qualifizierten Rollen

Qualifizierte Rollen haben einen Qualifier, aufgrund dessen Rollen gleicher Klassifizierung unterscheidbar gemacht werden, wie beispielsweise im durchgängigen Beispiel die Angestelltenrollen. Gleichzeitig kann bei qualifizierten Rollen eine Aussage zur Rollenidentität gemacht werden. Unterscheiden sich die qualifizierenden Attribute desselben Rollentyps in ihren Ausprägungen, so handelt es sich um unterschiedliche Rollen desselben Objektes aus der realen Welt. Haben die qualifizierenden Attribute die gleiche Ausprägung, so sind die Rollen identisch.

Unterstützung der Generalisierungsvererbung

In der objekt-orientierten Welt werden Rollen meist Top-Down, d.h. von der Entität zur Rolle, bzw. von der Superrolle zur Subrolle modelliert. Da im Web of Data schon viele Ressourcen definiert sind soll auch eine Bottom-Up Modellierung möglich sein, d.h. von der Subrollen zur Superrolle. Dazu ist es notwendig, dass Attribute von der Rolle zur Superrolle vererbt werden, quasi eine selektive Aufwärtsvererbung von Attributen mit der Möglichkeit die Beziehung zwischen den Attributen zu definieren.

4.3 Analyse der Rahmenbedingungen im Web of Data vs. objekt-orientierter Welt

Rollen sind ein wichtiges Modellierungskonzept. In der objekt-orientierten Welt ist man sich dessen bewusst, wenngleich es auch kein einheitliches Rollenmodell gibt [Ste00]. Obwohl die zuvor definierten Anforderungen zum Teil in bereits bestehenden Rollenmodellen gefunden werden können, ist aufgrund der unterschiedlichen Rahmenbedingungen keine triviale Umsetzung von objekt-orientierten Rollenmodellen in das Web of Data möglich. Die Anforderungen entsprechen zu einem großen Teil dem Rollenmodell von [GSR96]. Aus diesem Grunde soll dieses Rollenmodell als Basis herangezogen werden und die Unterschiede in den Rahmenbedingungen aufgezeigt werden. Tabelle 1 stellt die Rahmenbedingungen im Rollenmodells von [GSR96] den Rahmenbedingungen im Web of Data gegenüber.

Rahmenbedingungen in [GSR96]	**Rahmenbedingungen im Web of Data**
Closed World Assumption (CWA)	Open World Assumption (OWA)
Unique Name Assumption (UNA)	Non-Unique Name Assumption (NUNA)

einfache Klassenzugehörigkeit	mehrfache Klassenzugehörigkeit
einfache Klassenvererbung	mehrfache (zyklische) Klassenvererbung
Strukturvererbung	keine Strukturvererbung
strikte Schemaebene	lose Schemaebene
keine Laufzeitflexibilität	volle Laufzeitflexibilität
zentrale Steuerungsmechanismen	dezentrale Steuerungsmechanismen
Top-Down Ansatz	Bottom-Up, Top-Down Ansatz

Tabelle 1: Rahmenbedingungen in [GSR96] vs. Rahmenbedingungen im Web of Data

Closed World Assumption vs. Open World Assumption

Im Ansatz von [GSR96] geht man von der Closed World Assumption und somit von vollständigen Informationen aus. Die Closed World Assumption besagt, dass jede Aussage die nicht wahr ist, falsch ist. Im Web of Data geht man nicht davon aus, sondern man geht von unvollständigen Informationen und der Open World Assumption aus [KlCa04]. Diese besagt, dass nicht beweisbare Aussagen weder wahr noch falsch sind, d.h. es gibt nicht nur wahr und falsch, sondern auch unbekannt.

Unique Name Assumption vs. Non Unique Name Assumption

In der objekt-orientierten Welt und auch im Rollenmodell von [GSR96] geht man von der Unique Name Assumption aus. Demnach repräsentieren Objekte mit unterschiedlichen Identifiern auch unterschiedliche Objekte aus der realen Welt. Im Web of Data ist dies nicht gewährleistet. Wenn zwei Ressourcen durch zwei URIs unterschiedlich benannt sind gibt dies noch keinen Aufschluss darüber ob die URIs ein und dasselbe Objekt aus der realen Welt identifiziert oder nicht.

Klassifizierung: einfach vs. mehrfach

In den meisten Rollenmodellen, so auch in jenem von [GSR96], können Objekte direkt nur maximal einer Klasse angehören und indirekt über Subklassenbeziehungen mehreren Klassen. Die Zugehörigkeit ist unveränderlich und muss bei der Objektinstanzierung definiert werden. In RDF(S) ist dies gänzlich anders. Eine Ressource kann mehreren Klassen angehören. Dadurch, dass in einer Semantic Web Applikation laufend Daten nachgeladen werden können, kann sich die Klassenzugehörigkeit entwickeln und kann somit abhängig davon sein, welche Informationsressourcen gerade geladen sind. Einzig das monotone Folgern muss gewährleistet sein, d.h., dass die bestehende Klassifizierung einer

Ressource nicht durch neu geladene Informationen „gelöscht“ werden kann. Außerdem erfolgt eine Klassifizierung nicht nur direkt über rdf:type, sondern kann auch indirekt über Attribute erfolgen. [ODG+07] [OHD08]

Klassenvererbung: einfach vs. mehrfach (zyklisch)

Im Rollenmodell von [GSR96] kann eine Klasse nur von maximal einer einzigen Superklasse erben. In RDF(S) kann eine Klasse Subklasse von mehreren Superklassen sein. Die Klassenhierarchie kann sogar Zyklen enthalten. [ODG+07] [OHD08]

Strukturvererbung: Vererbung von Attributen und Methoden vs. keine Vererbung

In der objekt-orientierten Welt, so auch bei [GSR96] wird jedes Attribut, jede Methode lokal zu einer Klasse definiert. Subklassen dieser Klasse erben alle Attribute und Methoden und können deren Definition verfeinern und überschreiben. In RDF(S) gibt es keine Methoden. Attribute (Prädikate) werden unabhängig von Klassen definiert; die lose Zuordnung einer Klasse über eine Domainangabe entspricht nicht einer klassenlokalen Definition wie in der Objektorientierung, sondern führt zu einer entsprechenden Klassifizierung von Ressourcen. Es können alle Attribute (Prädikate) auf jede Ressource angewandt werden. Dementsprechend gibt es in RDF(S) keine Vererbung im Sinne der Objektorientierung. [ODG+07] [OHD08]

Schemaebene: strikt vs. lose

Bei [GSR96] sind Klassendefinitionen strikt, d.h. die Klassendefinition enthält alle für die Beschreibung relevanten Eigenschaften einer Instanz. Weiters ist eine Klassendefinition Voraussetzung, dass es Instanzen gibt. Instanzen entsprechen genau der definierten Struktur auf Schemaebene. In RDF(S) können Instanzen auch ohne Schemainformation existieren bzw. auch von der Schemainformation abweichen. [ODG+07] [OHD08]

Laufzeit Flexibilität: keine vs. volle

Im Rollenmodell von [GSR96] können sich Klassendefinitionen zur Laufzeit nicht ändern - zumindest wird auf solche Änderungen nicht eingegangen. Im Gegensatz dazu ist dies in RDF(S) durch die Integration von heterogenen Daten mit unterschiedlichen Strukturen möglich. Sowohl das Schema als auch die Instanzen können zur Laufzeit ergänzt werden, da, wie bereits unter dem Punkt Klassifizierung erläutert worden ist, Semantic Web Applikationen zur Laufzeit Daten aus dem Web nachladen können und dies, unter

Einhaltung des monotonen Folgerns, zu einer Ergänzung bereits bestehender Daten, so auch von Klassen, führen kann. [ODG+07] [OHD08]

Steuerungsmechanismen: zentrale vs. dezentrale

In der objekt-orientierten Welt gibt es zentrale Steuerungsmechanismen und es kann beispielsweise festgelegt werden, wer auf welche Daten Schreibrechte hat. Dadurch ist eine gewünschte Homogenität der Struktur und Daten gewährleistet. Das Web of Data ist dezentral, sowohl auf Schema- als auch auf Instanzebene, und es kann jeder über jeden Aussagen machen, wodurch sehr heterogene Strukturen und Daten entstehen können.

Vorgehensweise: Top-Down vs. Top-Down und Bottom-Up

Im Rollenmodell von [GSR96] ist die Vorgehensweise bei der Implementierung des Rollenmodells Top-Down, d.h. von den generellen Klassen bzw. Rollen zu den speziellen. Im Web of Data ist eine Top-Down Vorgehensweise nicht immer zweckmäßig, nämlich, wenn beispielsweise bereits existierende Beschreibungen von Nicht-Informationsressourcen eingebunden werden und aus diesen eine generelle Ressource erstellt werden soll (vgl. Kapitel 4.2.2.3). Aus dieser neu erstellten Ressource kann wiederum eine Spezialisierungs- oder auch eine Generalisierungsrolle erstellt werden.

4.4 Entwicklung eines Rollenmodells für das Web of Data

Nachdem nun die Anforderungen an ein Rollenmodell im Web of Data und die unterschiedlichen Rahmenbedingungen in der objekt-orientierten Welt und im Web of Data erläutert worden sind, wird nun ein Rollenmodell für das Web of Data erstellt. Dazu soll zuerst die bereits angesprochene getrennte Klassifizierung, nämlich auf Entitätsebene und auf Rollenebene erläutert werden. Nach Aussagen zu allgemeinen Eigenschaften von Rollen wird auf die drei verschiedenen Ausbaustufen von Rollen eingegangen:

a) simple Rollen,
b) qualifizierte Rollen und
c) simple und qualifizierte Rollen mit Generalisierungsinformationen.

Zum Ende dieses Unterkapitels wird auf Restriktionen im Zusammenhang von Rollen und RDF(S) eingegangen.

4.4.1 Zwei Identitätslevel: Entitätslevel und Rollenlevel

Um der Anforderung, Ontologien wieder zu verwenden, gerecht zu werden (vgl. Kapitel 4.3) und einen Beitrag zum Counting-Problem nach [WJS95] zu leisten (vgl. Kapitel 3.2.2), ist es notwendig, eine separate rollenspezifische Klassifizierung vorzunehmen und eine Nicht-Informationsressource auf zwei unterschiedlichen Level zu klassifizieren. Nämlich auf dem bereits in Kapitel 2.6.4 eingeführten Entitätslevel und auf dem fein granularen Rollenlevel.

In diesem Zusammenhang sei nochmals die Definition von Identität nach [Hal06] erwähnt: „The classic definition of identity is whether or not two objects are in fact, on some given level, the same.". Dies bedeutet, dass die Identität von zwei Objekten abhängig ist vom Betrachtungslevel.
Der erste Betrachtungslevel, der bereits in Kapitel 2.6.4 eingeführt worden ist, ist der Entitätslevel. Auf diesem Level werden alle darunter liegenden Objekte (Modelle, Beschreibungen) als identisch betrachtet. Das sind alle Rollen (vgl. Abbildung 20).
Der zweite Betrachtungslevel ist der Rollenlevel. Der Rollenlevel beinhaltet Beschreibungen bzw. Modelle einer Nicht-Informationsressource in einem speziellen Kontext. Durch die hierarchische Anordnung von Rollen (vgl. Abbildung 20) ergeben sich weitere Betrachtungslevels, auf diese wird aber in dieser Diskussion zu Identitätslevels nicht weiter eingegangen.

Würde es keine Trennung zwischen Entitätslevel und Rollenlevel geben, könnte es vorkommen, dass, was durch das Counting-Problem ausgedrückt wird, verletzt wird. Das Counting-Problem zeigt in dem Zusammenhang, dass die Anzahl an Instanzen, gezählt aufgrund ihrer Rollen, größer ist als dieselben Instanzen, gezählt aufgrund der Objekte die diese Rollen einnehmen [Ste00] [WJS95] (vgl. Kapitel 3.2.2). Beispielsweise kann sich die Anzahl der Passagiere, die von einem gewissen Transportmittel innerhalb einer Periode befördert werden, von der Anzahl der Personen, die vom selben Transportmittel in derselben Periode transportiert werden, unterscheiden. Der Grund ist, dass die Rolle „Passagier" von einer Person mehrfach eingenommen werden kann. Selbiges gilt, wenn die Anzahl der Projektmanager in einem Unternehmen gezählt werden soll. Diese Rolle kann von einem Angestellten mehrfach eingenommen werden. Somit kann die Anzahl der Projektmanagerrollen größer sein als die Anzahl der Angestelltenrollen.

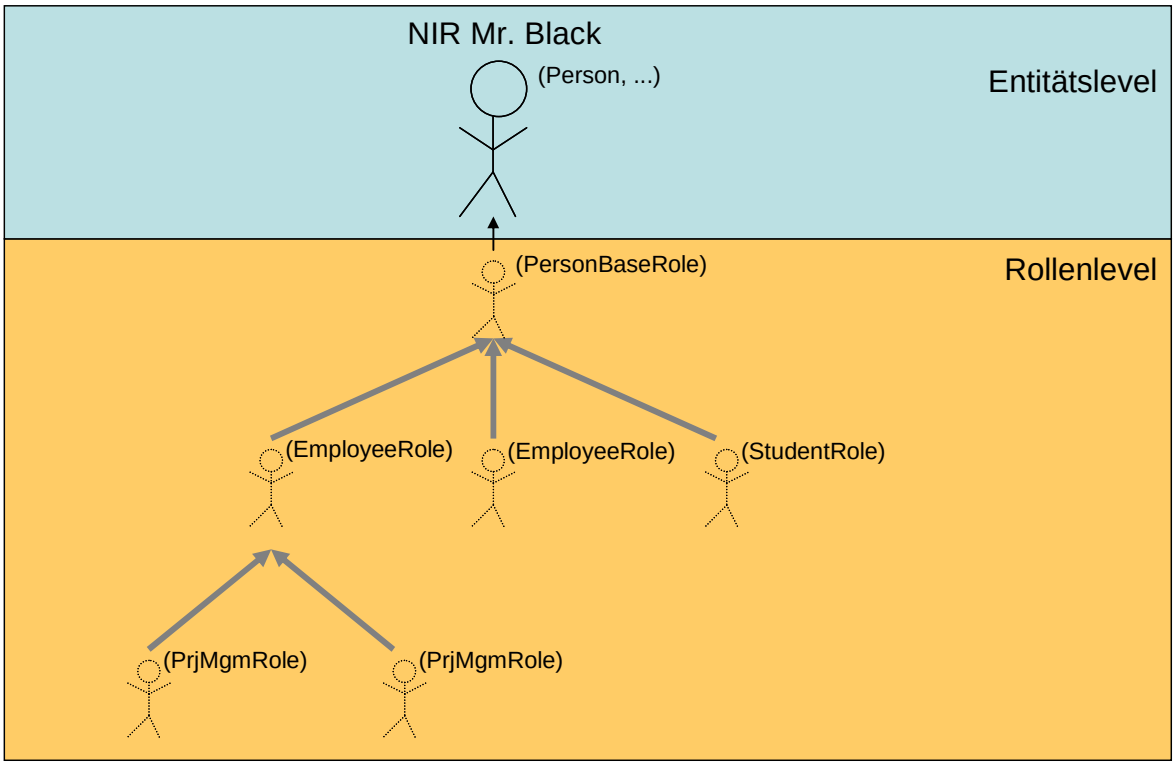


Abbildung 20: Entitätslevel und Rollenlevel

Würde nicht getrennt werden zwischen Klassifizierung auf Rollenebene und Klassifizierung auf Entitätsebene, könnte es zu einem Problem kommen. Dies entsteht, wenn eine Ressource mehrfach klassifiziert wird. Wie schon in Kapitel 2.5.4.1 erläutert kann in RDF(S) nicht nur über rdf:type, sondern auch über Domain- und Rangeangaben bei Prädikaten klassifiziert werden. Beispielsweise, wenn ein Prädikat „name“ als Domain „Person“ hat und dieses Prädikat auf eine Ressource, benannt mit der URI http://...#black, welche als Passagier klassifiziert ist, angewandt wird, dann ist die URI http://...#black sowohl als Person als auch als Passagier klassifiziert und stellt eine Verletzung des Counting-Problems dar. Denn eine Zählung von Passagieren und Personen würde nicht zum richtigen Ergebnis führen.

Die Klassifizierung im herkömmlichen Sinne, die in RDF(S) mittels Domain und Range über Attribute oder direkt mittels Typzuweisung erfolgt, hat keine Auswirkung auf die Klassifizierung auf Rollenebene und somit wird die Anforderung, dass Vokabulare wieder verwendet werden können, unterstützt.

4.4.2 Allgemeines zu Rollen im Web of Data

Wie bereits in Kapitel 4.1.1 erläutert, ist es notwendig mehrere Modelle als Stellvertreter eines Objektes aus der realen Welt zu erzeugen um Rollen im Web of Data abbilden zu können. Durch eine Rollenverlinkung der URIs mit deren Beschreibung diese Modelle entstehen, wird eine Rollenhierarchie (auf Instanzebene) gebildet, die entitätsspezifisch ist. In dieser Rollenhierarchie werden durch die Rollenverlinkung die Modelle (Rollen) so angeordnet, dass die allgemeine Rolle die Wurzel bildet und die speziellen Rollen die Blätter bilden. Ist eine Rolle in dieser Hierarchie auf Rollenebene nicht klassifiziert, so bedeutet dies, dass der Kontext in dem die Rolle gültig ist nicht angegeben ist. Die Rollenbeziehung besagt dann lediglich, dass eine Rolle eine Subrolle einer anderen Rolle ist und beide Rollen dieselbe Nicht-Informationsressource beschreiben.

Neben der Rollenhierarchie auf Instanzebene gibt es auch eine Rollenklassenhierarchie auf Schemaebene. Eine solche Hierarchie ist definiert durch Superrollen-Beziehungen zwischen Rollenklassen. Es handelt sich bei dieser Beziehung zwischen Rollenklassen, im Gegensatz zu Subklassenbeziehungen, um keine „is a“ Beziehung und somit findet auch keine Typenvererbung statt.

4.4.3 Simple Rollen

Simple Rollen sind ein sehr leichtgewichtiger Ansatz um Rollen im Web of Data zu implementieren. Die einfachste Form einer simplen Rolle entsteht durch eine Rollenverlinkung von zwei URIs, die beide dieselbe Ressource identifizieren. Dadurch wird ausgedrückt, dass beide URIs dieselbe Nicht-Informationsressource identifizieren. Weiters werden dadurch die Rollen hierarchisch geordnet und ausgedrückt, dass sie sich auf Kontextebene unterscheiden. D.h. eine Rolle wird zur Superrolle der anderen Rolle bzw. vice versa, eine Rolle zur Subrolle der anderen Rolle und die Superrolle beschreibt die gemeinsame Nicht-Informationsressource in einem anderen Kontext als die Subrolle. Die Ausdrucksstärke einer simplen Rolle kann mittels Klassifizierung auf Rollenebene noch verstärkt werden. Dadurch wird dann der Kontext der Rolle definiert, beispielsweise als Privatperson oder Student.

Eine simple Rolle ist jedoch problematisch, wenn sie mehrfach von einer Nicht-Informationsressource eingenommen wird. Dann kann (im Gegensatz zu qualifizierten Rollen) nicht mehr unterschieden werden, in welchem speziellen Kontext die Rolle gültig ist. Weiters können über simple Rollen (im Gegensatz zu qualifizierten Rollen), welche dieselbe Ressource identifizieren und nicht durch ein Rollenstatement miteinander verlinkt

sind (direkt oder indirekt), aufgrund der Non-Unique Name Assumption keine Aussagen darüber gemacht werden, ob sie identisch sind oder nicht (vgl. Kapitel 4.4.1).

Abbildung 21 bezieht sich auf das durchgängige Beispiel aus Kapitel 1.2. In diesem Beispiel werden die allgemeine Rolle (PersonBaseRole) und die Angestelltenrollen (EmployeeRole) mittels simpler Rollen dargestellt. Die URIs werden entsprechend mit Rollenstatements miteinander verknüpft, sodass sich die Modelle in einer Rollenbeziehung zueinander befinden. Die URI <http://...#black> beschreibt die Nicht-Informationsressource Mr. Black mit seinen allgemein gültigen Attributen und das Modell von Mr. Black, das dadurch entsteht, ist auf Rollenebene als PersonBaseRole klassifiziert. Die URIs <http://...#empblack1> und <http://...#empblack2> beschreiben Mr. Black in der Rolle als Angesellter. Die dadurch entstehenden Modelle werden auf Rollenebene als EmployeeRole klassifiziert und sind Subrollen der allgemeinen Rolle (PersonBaseRole). D.h. jenes Modell, das als PersonBaseRole klassifiziert ist, ist die Superrolle der beiden Rollen, welche als EmployeeRole klassifiziert sind. Im Unterschied zum Rollenmodell von [GSR96] werden hier auf RDF-Ebene keine Prädikate von oben nach unten propagiert. Falls dies gewünscht ist, kann es vom Interpreten gemacht werden.
Durch die Rollenbeziehung wird ausgedrückt, dass die Angeselltenrollen (klassifiziert als EmployeeRole) und die allgemeine Rolle (klassifiziert als PersonBaseRole) Beschreibungen von Mr. Black in einem unterschiedlichen Kontext sind. Sie sind also auf Rollenebene unterschiedlich. Dadurch, dass die beiden Angestelltenrollen in keiner Rollenbeziehung zueinander sind und im Web of Data die Unique Name Assumption nicht gilt, kann keine Aussage darüber gemacht werden, ob diese beiden Rollen identisch sind oder nicht.

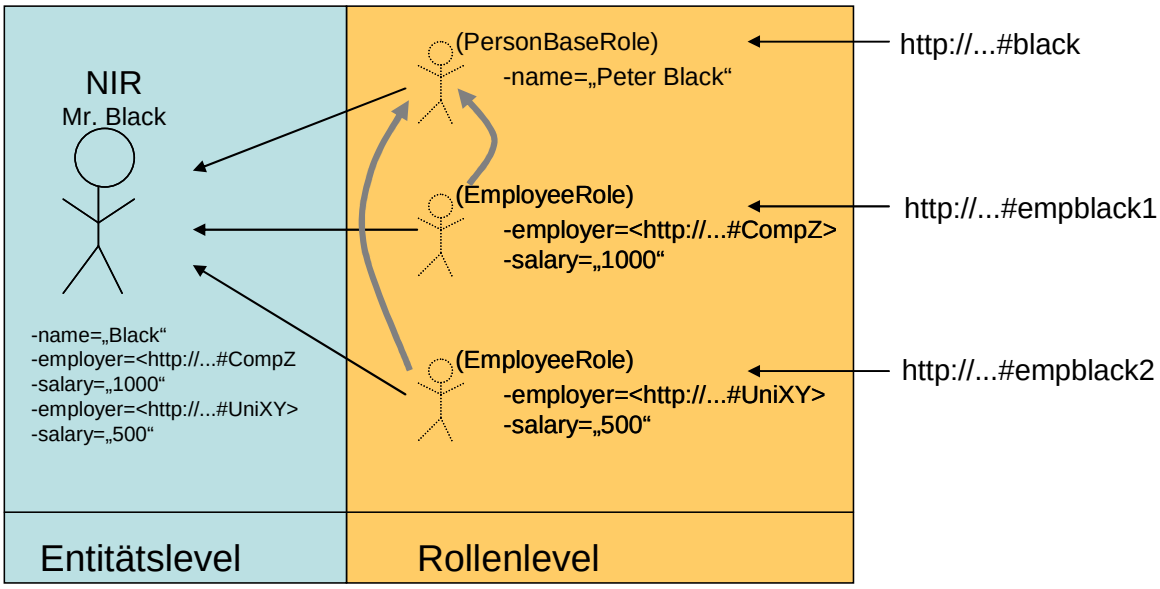


Abbildung 21: Simple Rollen

4.4.4 Qualifizierte Rollen

Qualifizierte Rollen sind einer Erweiterung der simplen Rollen. Die Erweiterung beinhaltet die Möglichkeit qualifizierende Attribute für einen Rollentyp zu definieren. Unterscheiden sich die Ausprägungen der qualifizierenden Attribute zweier Rollen gleichen Rollentyps, so besagt dies, dass die Rollenbeschreibungen in einem unterschiedlichen Kontext sind.

Abbildung 22 zeigt dieselbe Situation wie Abbildung 21, jedoch mit qualifizierenden Attributen. Diese sind fett und unterstrichen dargestellt. Die qualifizierenden Attribute ermöglichen eine Unterscheidung von Ausprägungen desselben Rollentyps und definieren den Gültigkeitsbereich bzw. den genauen Kontext in dem die Rolle gültig ist. D.h. <http://...#empblack1> ist eine Beschreibung von Mr. Black im Kontext als Angestellter bei der Firma CompZ und <http://...#empblack2> ist eine Beschreibung von Mr. Black im Kontext als Angestellter, allerdings bei der Firma UniXY.

Im Web of Data gilt grundsätzlich die Non-Unique Name Assumption und es kann aufgrund unterschiedlicher Namen (URIs) keine Aussage über die Identität gemacht werden. Unterscheiden sich jedoch die qualifizierenden Attribute in ihren Ausprägungen von Rollen derselben Entität und desselben Rollentyps, so kann daraus gefolgert werden, dass die Rollen einen unterschiedlichen Gültigkeitsbereich haben und deshalb nicht identisch sind. Beispielsweise kann eine Aussage für <http://...#empblack1> und

<http://...#empblack2> getätigt werden, die besagt, dass diese URIs dieselbe Nicht-Informationsressource in einem unterschiedlichen Kontext beschreiben und die Modelle, als Stellvertreter der Nicht-Informationsressource, in einem unterschiedlichen Kontext gültig sind.

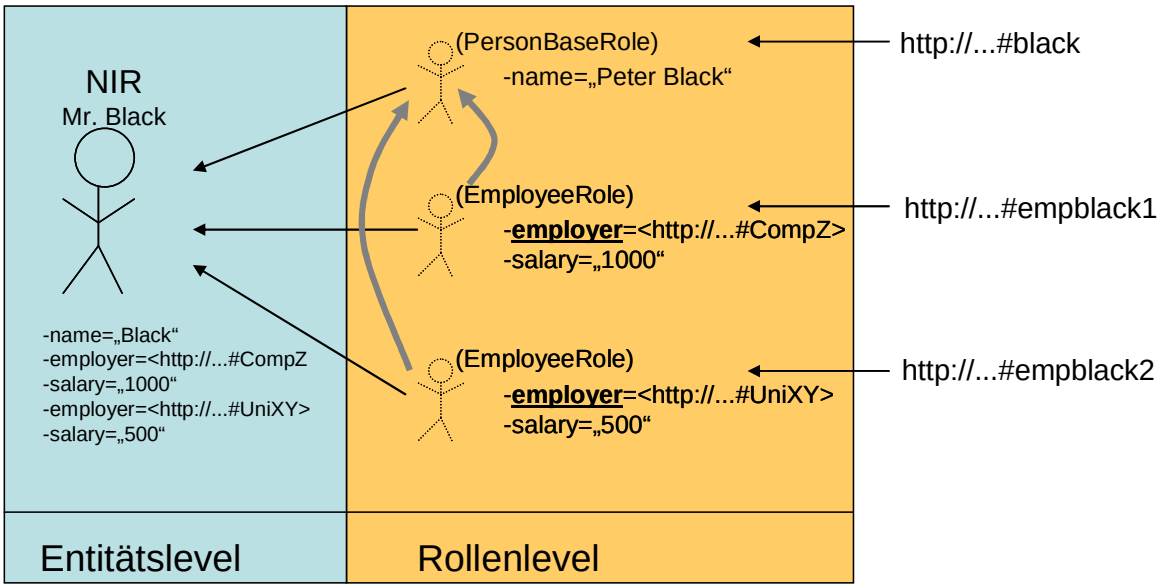


Abbildung 22: Qualifizierte Rollen

4.4.5 Simple und qualifizierte Rollen mit Generalisierungsinformationen

Eine Erweiterung von simplen und qualifizierten Rollen ist die Möglichkeit Generalisierungsinformationen, in der Form von Informationen über die Aufwärtsvererbung von Prädikaten und deren Ausprägung, zu definieren.

Dadurch, dass im Web of Data eine Nicht-Informationsressource von mehreren unterschiedlichen URIs identifiziert werden kann, d.h. unabhängig voneinander eine Vielzahl an Beschreibungen desselben Objektes aus der realen Welt entstehen können, kann man nicht immer von einem Top-Down Ansatz ausgehen. In Kapitel 4.2.2 im Anwendungsfall drei, bei dem es um die Erstellung einer Generalisierungsrolle geht, ist dies der Fall. In diesem Szenario existieren zuerst die Rollen und auf Basis dieser soll eine Superrolle definiert werden, wobei eine Aufwärtsvererbung von Prädikaten beschrieben werden soll. Generalisierungsinformationen sollen nicht nur Aufwärtsvererbungen auf Instanzebene beschreiben, sondern auch auf Schemaebene. Unter einer Beschreibung einer Aufwärtsvererbung auf Instanzebene wird verstanden, dass eine Rollenvererbung von einer Rolle zu einer in der Rollenhierarchie darüber liegenden Rolle (Superrolle) beschrieben

wird. Eine Beschreibung einer Vererbung auf Schemaebene bezieht sich nicht auf einzelne Rollen, sondern auf die Extension von Rollenklassen.

Eine Generalisierungsinformation, die sich auf die Instanzebenen bezieht, könnte beispielsweise ausdrücken, dass von einer Rolle identifiziert durch URI 1 das Prädikat Name mit seiner Ausprägung zur Superrolle identifiziert durch URI 2 vererbt werden soll. Eine Generalisierungsinformation die sich auf die Schemaebene bezieht, könnte beispielsweise ausdrücken, dass entitätsspezifisch von jeder Rolle, die auf Rollenebene als EmployeeRole klassifiziert ist, das Prädikat Name zu jeder (direkten oder indirekten) Superrolle, die auf Rollenebene als PersonBaseRole klassifiziert ist, vererbt werden soll.

In Abbildung 23 wird ein Ausschnitt aus dem durchgängigen Beispiel aus Kapitel 1.2 dargestellt und davon ausgegangen, dass zuerst die Beschreibungen mit den URIs <http://...#empblack1> und <http://...#empblack2> existieren. Beide sind unabhängige autoritative Beschreibungen in unterschiedlichen Namensräumen. Mr. Black hat keine Kontrolle über diese Namensräume und möchte nun eine allgemeine Beschreibung über sich selbst erzeugen und die Attribute aus bereits existierenden Beschreibungen von ihm wieder verwenden. Dazu nutzt er die Möglichkeit Generalisierungsinformationen zu erstellen. In Abbildung 23 sind die aufwärts vererbten Attribute fett und kursiv dargestellt.

Die Generalisierungsinformationen sollen nicht nur beschreiben welche Attribute aufwärts vererbt werden sollen, sondern auch in welcher Beziehung die zu vererbenden Attribute zueinander stehen. [ScNe88] haben sich mit Generalisierungsbeziehungen in der objekt-orientierten Welt beschäftigt und haben mögliche Beziehungen zwischen aufwärts zu vererbenden Attributen analysiert. Unter der Voraussetzung, dass zwei textuell gleiche Prädikate aufwärts vererbt werden, können drei häufig vorkommende Beziehungen identifiziert werden.

a) identische Attribute

   Die Attribute zweier Rollen sind identisch, wenn sie dasselbe Attribut des Objektes aus der realen Welt beschreiben und dieses nicht rollenspezifisch ist. In Abbildung 23 sind dies die Attribute name und ssnr bzw. ID. Werden die Attribute aufwärts vererbt und haben wider Erwarten unterschiedliche Ausprägungen, so wird jede Ausprägung nach oben vererbt, ansonsten wird die gemeinsame Ausprägung vererbt.

b) Rollenbeziehung

Zwei Attribute zweier Rollen sind in einer Rollenbeziehung, wenn sie dasselbe Attribut des Objektes aus der realen Welt beschreiben und dieses rollenspezifisch ist. D.h. jedes der beiden Attribute stellt einen Teilaspekt dar. In Abbildung 23 ist dies beim Attribut salary der Fall und die Summe der beiden Attribute ergibt das nach oben zu vererbende Attribut. Die Summe der Attribute kann entweder durch eine mathematische Aufsummierung erzeugt werden, oder bei alphanumerischen Ausprägungen durch eine Vererbung der jeweiligen Ausprägungen.

c) Gegenpart-Beziehung

Zwei Attribute zweier Rollen befinden sich in einer Gegenpart-Beziehung, wenn sie dasselbe Attribut des Objektes aus der realen Welt beschreiben und dieses ausschließlich in der spezifischen Rolle von Relevanz ist.

Gerade im Web of Data kann es durch die Vielzahl an Vokabularen vorkommen, dass zwei textuell unterschiedliche Prädikate (Vokabel), semantisch dieselbe Eigenschaft, desselben Objektes aus der realen Welt beschreiben. Dies ist beispielsweise der Fall, wenn zwei Attribute unterschiedliche Namensräume haben oder von unterschiedlichen natürlichen Sprachen sind, aber dieselbe Eigenschaft beschreiben. Deshalb können zwei (oder mehrere) unterschiedliche Vokabel zu einem Vokabel zusammengefasst werden und aufwärts vererbt werden.

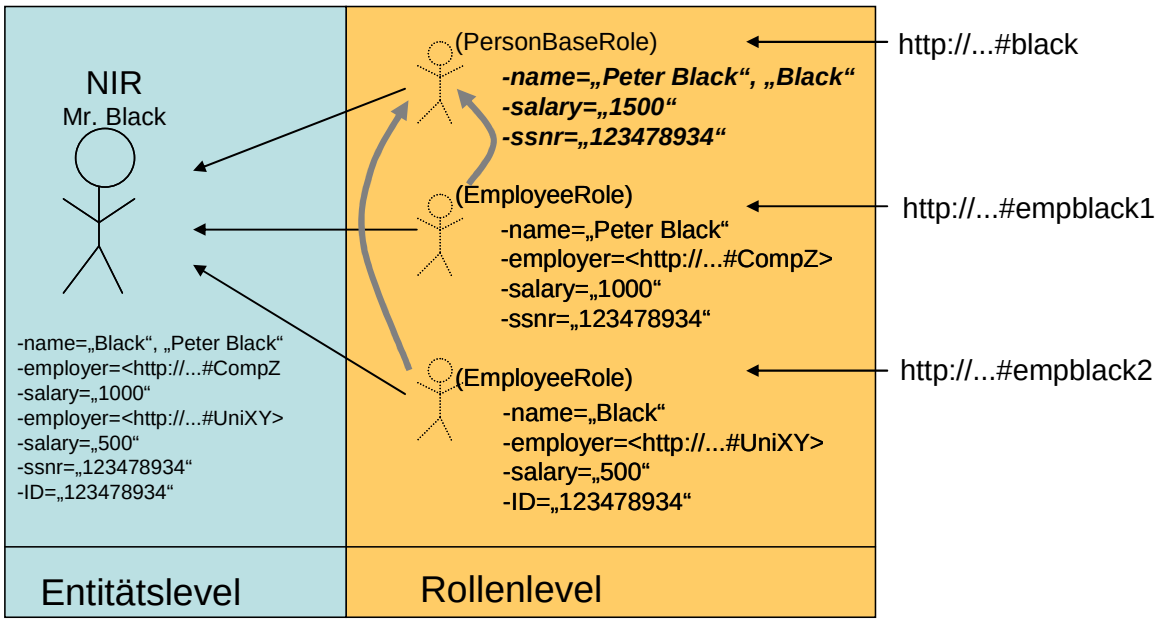


Abbildung 23: Simple Role mit Aufwärtsvererbung

4.4.6 Restriktionen für Rollen

Im Rollenmodell von [GSR96] können Rollen nur in Verbindung mit einer Entität existieren, d.h. es wird explizit zwischen dem Entitätsobjekt und den vereinnahmten Rollen unterschieden. Dies macht es möglich, dass Rollenobjekte ausschließlich kontextspezifische Attribute enthalten und alle allgemeinen Attribute bei dem Entitätsobjekt stehen. Im Web of Data muss man aufgrund der geltenden Rahmenbedingungen (vgl. Kapitel 4.3) von unvollständigen Informationen ausgehen. Außerdem ist es in RDF(S) nicht möglich derartige Auflagen zu formulieren und es kann nicht garantiert werden, dass zu einer Rolle immer auch eine Entitätsressource verfügbar wäre. Aus diesem Grunde sind Rollen im Web of Data eigenständig.
Was im Rollenmodell von [GSR96] einer Entität entspricht, ist hier eine Rolle, die in einem allgemeinen Kontext beschrieben wird. Ein weiterer Grund für die Nichtunterscheidung zwischen Entität und Rolle im Rollenmodell für Web of Data ist, dass es innerhalb der Rollenhierarchie zu Generalisierungen und Aufwärtsvererbungen kommen kann.
Weiters können im Rollenmodell von [GSR96] nur Objekte mit einem bestimmten Typ Rollen annehmen. Beispielsweise kann nur eine Person eine Angestelltenrolle einnehmen und nur Angestellte können eine Projektmanagerrolle einnehmen. Im Web of Data bzw. in RDF(S) können solche Restriktionen nicht definiert werden. Es besteht die Möglichkeit, dass Subrollen ohne Superrolle existieren. Es können beispielsweise Projektmanagerrollen eines Objektes aus der realen Welt existieren, ohne dass eine Angestelltenrolle existiert bzw. sogar, ohne dass irgendeine Ressource existiert, die auf dasselbe Objekt aus der realen Welt verweist.

Die einzige Art und Weise, wie Restriktionen im Rollenmodell definiert werden können, ist mit Hilfe der qualifizierenden Attribute. Ist das qualifizierende Attribut beispielsweise nur die Rollenbeziehung, so kann damit sichergestellt werden, dass diese Rolle nur einfach von deren Superrolle eingenommen werden kann. Beispielsweise, wenn das qualifizierende Attribut der PrivatePersonRole die Rollenbeziehung zur PersonBaseRole ist, so ist sichergestellt, dass Mr. Black in der allgemeinen Rolle (PersonBaseRole) diese Rolle nur einfach annehmen kann. Existieren mehrere Privatpersonenrollen als Subrolle der allgemeinen Rolle von Mr. Black, so sind diese identisch.

4.5 Das Rollenmodell – Beurteilung und Vergleich

Im Folgenden werden Rollen im Web of Data mit Rollen in der objekt-orientierten Welt verglichen. Weiters wird die Eignung von Rollen im Hinblick auf Navigierbarkeit und Lösung für das Coreferenz-Problem beurteilt.

4.5.1 Rollen im Web of Data vs. Rollen in [GSR96]

Neben den Unterschieden, die sich aufgrund der Rahmenbedingungen ergeben (vgl. Kapitel 4.3), hat sich im Laufe der Entwicklung des Rollenmodells für das Web of Data ein weiterer wesentlicher Unterschied ergeben: Rollen im Web of Data können, im Gegensatz zu [GSR96], auch ohne ein Entitätsobjekt existieren, bzw. wird nicht unterschieden zwischen einem Entitätsobjekt und einer Rolle. Was bei [GSR96] das Entitätsobjekt, also das Objekt mit der allgemeinen Beschreibung der Entität ist, ist in unserem Ansatz eine Rolle in einem allgemeinen Kontext. Der Grund dafür ist, dass der Ansatz von [GSR96] ein Top-Down Ansatz ist, während der Ansatz für das Web of Data ein gemischter Ansatz ist, sowohl Top-Down als auch Bottom-Up. Dadurch können vermeintliche Superrollen zu Subrollen werden. Um dies zuzulassen wird kein „allgemeinstes“ Objekt als Entitätsobjekt definiert und diese Unterscheidung von [GSR96] wird aufgebrochen.

4.5.2 Navigierbarkeit zwischen Rollen

Wesentlich für die Navigierbarkeit von Rollen ist die Positionierung des Rollenstatements. Findet die gesamte Beschreibung der Rollen in einer einzigen Informationsressource statt (vgl. Abbildung 24a), so kann beliebig zwischen den Rollen navigiert werden. Da im Web of Data aber Aussagen über eine Nicht-Informationsressource beliebig über mehrere Files aufgeteilt sein können und das Rollenstatement die Verbindung zwischen den beschriebenen Ressourcen in diesen Files darstellt, gibt die Positionierung des Rollenstatements Aufschluss darüber, wie zwischen den einzelnen Rollen navigiert werden kann. Bei einer Verlinkung zwischen zwei Rollen gibt es mehrere mögliche Positionen an denen das Rollenstatement angebracht werden kann: bei der Superrolle, bei der Subrolle, an dritten unabhängigen Position oder redundant an den jeweiligen Positionen. Auf redundante Rollenstatements sollte jedoch verzichtet werden, da in vielen Applikationen gleiche Statements in unterschiedlichen Informationsressourcen als zwei unterschiedliche Statements aufgefasst werden. Besser ist ein Verweis mittels rdfs:seeAlso von einer Informationsressource zu einer anderen, welcher besagt, dass über die Subjektressource in

der Objektressource weitere Informationen gefunden werden können [MaMi04a] [MaMi04b].

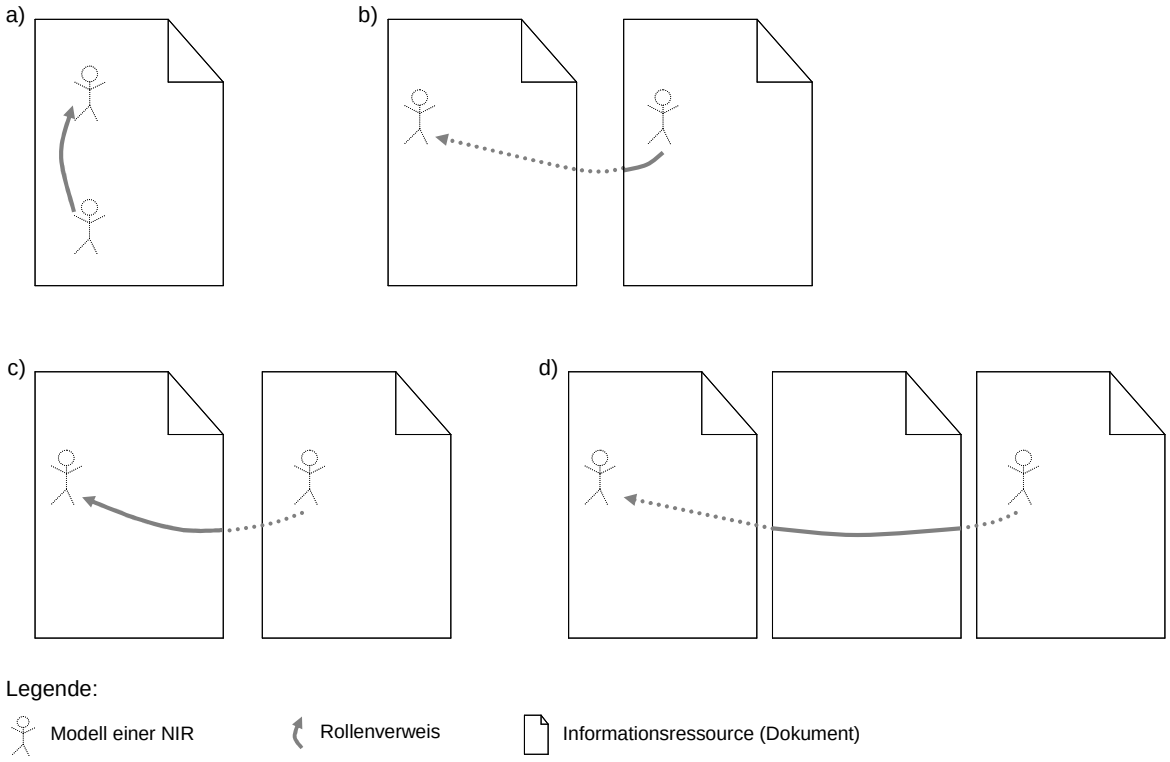


Abbildung 24: Möglichkeiten der Aufteilung einer Rollenbeziehung (vgl. Abbildungen 15-19)

Steht das Rollenstatement ausschließlich bei der Subrolle (vgl. Abbildung 24b) und gibt es auch keinen rdfs:seeAlso Verweis von der Superrolle zur Subrolle, so kann ausschließlich von der Subrolle zur Superrolle navigiert werden. Das kommt einer Situation gleich, wo die Superrolle nichts von einer Subrolle weiß. Mit einem einfachen rdfs:seeAlso Verweis von der Superrolle zur Subrolle könnte dieses Problem behoben werden. Weiters würde dies nicht nur der Navigierbarkeit dienen, sondern es könnte auch zur Vertrauensevaluierung genutzt werden. Im Sinne dass, wenn die Superrolle einen Verweis zur Subrolle macht, diese von den Aussagen weiß und mit diesen auch einverstanden sind, d.h. diese auch vertrauenswürdig sind.

Steht das Rollenstatement ausschließlich bei der Superrolle (vgl. Abbildung 24c) und es gibt keinen rdfs:seeAlso Verweis von der Subrolle zur Superrolle, so entsteht dasselbe Problem wie vorhin beschrieben, nur vice versa. Dementsprechend kann es auch, wie oben erklärt, gelöst werden.

Steht das Rollenstatement an einer dritten unabhängigen Position (vgl. Abbildung 24d), beispielsweise in einem zentralen Rollenfile und es existieren keine Verweise, weder von der Superrolle noch von der Subrolle, so kann gar nicht zwischen den beiden navigiert werden. Mit einem rdfs:seeAlso Verweis auf das zentrale Rollenfile wäre das Problem schnell gelöst. Oftmals kann es allerdings gewollt sein, die identitätsspezifische Beziehung zwischen den Ressourcen nicht statisch einzubinden, sondern eventuell dynamisch zur Laufzeit. Diese Einbindung bewirkt dann eine Anreicherung der Daten um Rollenbeziehungen (entitätsspezifische Bündelung).

4.5.3 Coreferenz-Problem

Wie bereits gezeigt worden ist, liefert ein Rollenmodell für das Web of Data einen Lösungsbeitrag zum Coreferenz-Problem, welches eine Situation beschreibt in der mehrere URIs ein und dasselbe Objekt aus der realen Welt benennen [JGH08] [JGH07]. Während die Semantik von owl:sameAs zu stark ist, der Ansatz von [JGH08] und [JGH07] keine Möglichkeit den Kontext einer Beschreibung zu definieren bietet und der Ansatz von [BSCT08] den Kontext von Beschreibungen gar keine Beachtung schenkt, bietet das Rollenmodell eine flexible Lösung, die die Nachteile all dieser Ansätze eliminiert. Das Rollenmodell kann sowohl verteilt und dezentral von jedem Benutzer verwendet werden, als auch zentral, im Sinne eines zentralen Rollenfiles, das alle Rollenbeziehungen und -definitionen beinhaltet. Nicht nur diesbezüglich ist das Rollenmodell vollkommen flexibel, sondern auch was die Anzahl an notwendigen Definitionen anbelangt. Das Coreferenz-Problem kann bereits mit einem einzigen Link, nämlich mit einem Rollenverweis gelöst werden. Eine Klassifizierung oder weitere Definitionen sind keine Voraussetzung dafür. Weiters ist die Ausdrucksmächtigkeit und Leichtgewichtigkeit im Vergleich zu anderen Ansätzen bemerkenswert hoch.

Das Rollenmodell stellt beide Parteien zufrieden, nämlich jene, die für eine kontextspezifische Trennung von Modellen sind und jene, die dagegen sind. Erstere können die Rollenverlinkung als solche interpretieren und der Kontext geht nicht verloren. Letztere können eine Rollenbeziehung auf Entitätsebene interpretieren, auf der es dem Prädikat owl:sameAs entspricht.

4.5.4 Vordefinierte Rollenklassenhierarchien

Das bei der Beschreibung einer Nicht-Informationsressource verwendete Vokabular kann unter Umständen entscheiden, ob die URI als Name für die Nicht-Informationsressource in einer bestimmten Applikation verwendet werden kann oder nicht. [Boo06] meint: „… if I give you descriptive information that you cannot relate to anything else you already know, then you will not be able to make use of the URI, even if that descriptive information may be perfectly adequate for some other application“. Dies unterstreicht die Wichtigkeit von globalen Vokabularen. Deshalb erscheint es auch wichtig, dass es inhaltsspezifische Rollenklassenhierarchien gibt, um die Wiederverwendung dieser zu fördern. Ebenso wie es ein Vokabular zur Beschreibung von Personen und Beziehungen zwischen diesen gibt, wäre es sinnvoll und wichtig ebenso ein Standardvokabular für Rollen die Personen einnehmen können, zu definieren. Dies würde bei einer Generalisierung mit Generalisierungsinformationen (vgl. Kapitel 4.4.5), welche auf Schemaebene definiert ist, die Mächtigkeit dieser erhöhen. Beispielsweise könnte so definiert werden, dass jedes Vorkommen des Prädikates foaf:name innerhalb der Rollenhierarchie, entitätsspezifisch nach oben zur PersonBaseRole vererbt wird.

5. Rollen im Web of Data – Umsetzung mit RDF(S)

Nach der Entwicklung eines Rollenmodells für das Web of Data soll im Folgenden gezeigt werden, wie die vier Anwendungsfälle, definiert im Kapitel 4.2.2, im Web of Data umgesetzt werden können. Zuerst wird ein Rollenstandardvokabular definiert, welches Voraussetzung dafür ist, dass Rollen im Web of Data abgebildet werden können. Danach werden Ausschnitte aus den Anwendungsfällen umgesetzt und in Turtle Syntax dargestellt.

5.1 Rollenvokabular

Ein einheitliches Vokabular ist Voraussetzung dafür, dass Rollen von anderen Benutzern im Web of Data verwendet werden können und es zu anderen Ontologien in Relation gesetzt werden kann. [Boo06] bringt die Wichtigkeit eines Vokabulars anhand eines Beispiels zum Ausdruck und meint: „… if you receive a set of assertions expressed using the FOAF ontology, but you are unable to relate that ontology to the ontology you are using, the URI will be meaningless: you will not be able to make use of it as a proxy for its associated resource.“ Aus diesem Grund wird das folgende Rollenstandardvokabular eingeführt. Hinweis: Eine Definition der Semantik im Sinne einer Axiomatisierung des Rollenvokabulars ist nicht Teil dieser Arbeit.

Zuerst wird ein Rollenvokabular eingeführt um simple und qualifizierte Rollen im Web of Data abbilden zu können. Danach wird ein Rollenvokabular, um Generalisierungsinformationen in Form von Informationen über Aufwärtsvererbungen von Prädikaten beschreiben zu können, eingeführt, wobei zwischen Prädikaten zur Beschreibung der Aufwärtsvererbung auf Instanzebene und auf Schemaebene unterschieden wird. Das vollständige Rollenvokabular in RDF/XML Syntax kann dem Anhang entnommen werden.

5.1.1 Rollenvokabular für simple und qualifizierte Rollen

Im Folgenden wird nun Schritt für Schritt in das Standardrollenvokabular zur Abbildung von simplen und qualifizierten Rollen im Web of Data eingeführt.

Um im Web of Data eine Rollenbeziehung auszudrücken, werden zwei Modelle, die durch eine kontextspezifische Beschreibung mit einer URI entstehen, in eine Rollenbeziehung zueinander gesetzt. Das Prädikat, dass die beiden URIs verlinkt, ist rrdf:roleOf.

rrdf:roleOf

Das Prädikat rrdf:roleOf ist eine Instanz von rdf:Property und wird benutzt um auszudrücken, dass die Subjekt-URI und die Objekt-URI dieselbe Nicht-Informationsressource identifizieren und diese in einem unterschiedlichen Kontext beschreiben. Das Prädikat rrdf:roleOf ist functional, irreflexiv und azyklisch. Hinweis: Diese Eigenschaften können mit RDF(S) nicht ausgedrückt werden und sind deshalb hier nur als Kommentar zu verstehen.

Ein Tripple der Form:

R1 rrdf:roleOf R2

besagt, dass R1 eine Rolleninstanz einer Instanz von rrdf:RoleClass ist, R2 eine Rolleninstanz einer Instanz von rrdf:RoleClass ist und dass R2 Subrolle von R1 ist.

rrdf:specRoleOf, rrdf:genRoleOf

Die Prädikate rrdf:specRoleOf und rrdf:genRoleOf sind Subprädikate von rrdf:roleOf und werden verwendet um eine Rollenbeziehung zwischen zwei Ressourcen auszudrücken und anzumerken, dass es sich um eine Spezialisierungsrolle bzw. eine Generalisierungsrolle handelt. Es besteht kein semantischer Unterschied zu rrdf:roleOf, sie haben lediglich eine kennzeichnende Funktion um die Chronologie der Rollenerstellung festzuhalten.

Weiters werden für das Rollenmodell im Web of Data folgende Klassen benötigt:

rrdf:RoleClass

rrdf:RoleClass ist die Klasse aller Rollenklassen (Rollentypen).

rrdf:Role

Die Klasse rrdf:Role ist eine Instanz von rrdf:RoleClass. rrdf:Role ist die Superklasse aller Rollen.

Um eine Rollenklassenhierarchie, d.h. eine Hierarchie von Rollentypen zu bilden, wird das Prädikat rrdf:roleSuperClass verwendet:

rrdf:roleSuperClass

Das Prädikat rrdf:roleSuperClass ist eine Instanz von rdf:Property und drückt eine Spezialisierung bzw. Generalisierung zwischen zwei Instanzen von rrdf:RoleClass aus. rrdf:roleSuperClass ist functional, irreflexiv und azyklisch.

Ein Tripple der Form:

C1 rrdf:roleSuperClass C2

besagt, dass C1 eine Instanz von rrdf:RoleClass ist und C2 eine Instanz von rrdf:RoleClass ist und dass C1 eine speziellere Rollenklasse ist als C2. rdf:domain und rdf:range von rrdf:roleSuperClass ist rrdf:RoleClass.

Um eine Rolle (eine Rolleninstanz einer Instanz von rrdf:RoleClass) auf Rollenebene zu klassifizieren wird rrdf:roleType verwendet:

rrdf:roleType

Das Prädikat rrdf:roleType ist eine Instanz von rdf:Property und wird benutzt um ein Modell einer Nicht-Informationsressource (eine Rolle einer Entität) auf Rollenebene zu klassifizieren.

Ein Tripple der Form:

A rrdf:roleType R

besagt, dass R eine Instanz von rrdf:RoleClass ist und A auf Rollenebene eine Rolleninstanz von R.

rdf:range von rrdf:roleType ist rrdf:RoleClass.

Zur Definition von qualifizierenden Prädikaten für Rollenklassen wird owl:hasKey verwendet:

owl:hasKey

Das Prädikat owl:hasKey wird von OWL übernommen und wird benutzt um einer Instanz von rrdf:RoleClass Schlüsselattribute zuzuweisen.

Ein Tripple der Form:

R1 owl:hasKey (k1,k2)

besagt, dass R1 eine Instanz von rrdf:RoleClass ist, k1 und k2 Instanzen von rdf:Property sind und dass k1 und k2 Schlüsselprädikat einer Rolleninstanz von R1 sind.

5.1.2 Rollenvokabular für Generalisierungsinformationen auf Instanzebene

Um eine Generalisierungsvererbung auf Instanzebene beschreiben zu können, wird ein geeignetes RDF(S) Vokabular benötigt. Eine vollständige Beschreibung einer Generalisierungsvererbung wird durch fünf verschiedene Prädikate ausgedrückt, mit denen

die erbende Rolle, die Erblasserrolle, die Erblassprädikate, die erbenden Prädikate und die Beziehung zwischen den Erblassprädikaten definiert werden. Für Beispiele sei auf Kapitel 5.2.4 verwiesen. Für die Erklärung der Begrifflichkeiten sei auf Abbildung 25 verwiesen.
Um Generalisierungsinformationen auf Instanzebene erstellen zu können wird folgendes Vokabular eingeführt:

rrdf:player

Das Prädikat rrdf:player ist eine Instanz von rdf:Property und gibt den Erben der Generalisierungsvererbung an.
Ein Tripple der Form:

```
_:n rrdf:player P
```

besagt, dass P eine Rolleninstanz einer Instanz von rrdf:RoleClass ist und die erbende Rolle ist.

rrdf:role

Das Prädikat rrdf:role ist eine Instanz von rdf:Property und gibt die Erblassrollen der Generalisierung an.
Ein Tripple der Form:

```
_:n rrdf:role E
```

besagt, dass E eine Rolleninstanz einer Instanz von rrdf:RoleClass ist und die Erblassrolle ist.

rrdf:playerProperty

Das Prädikat rrdf:playerProperty ist eine Instanz von rdf:Property und gibt das erbende Prädikat der Generalisierungsvererbung an.
Ein Tripple der Form:

```
_:n rrdf:playerProperty A
```

besagt, dass A ein rdf:Property ist zugleich das erbende Prädikat ist. rdf:range von rrdf:playerProperty ist rdf:Property.

rrdf:roleProperty

Das Prädikat rrdf:roleProperty ist eine Instanz von rdf:Property und gibt das Erblassprädikat der Generalisierungsvererbung an.
Ein Tripple der Form:

```
_:n rrdf:roleProperty A
```

besagt, dass A ein rdf:Property ist zugleich das Erblassprädikat ist. rdf:range von rrdf:roleProperty ist rdf:Property.

rrdf:propertyRelation

Das Prädikat rrdf:propertyRelation ist eine Instanz von rdf:Property und gibt die Beziehung zwischen den Erblassprädikaten an.

Ein Tripple der Form:

```
_:n rrdf:propertyRelation R
```

besagt, dass R eine rrdf:Relation ist und die Beziehung zwischen den Erblassprädikaten definiert. rdf:range von rrdf:propertyRelation ist rrdf:Relation.

rrdf:Relation

Die Klasse rrdf:Relation ist die Klasse aller Beziehungen zwischen Erblassprädikaten.

rrdf:IdentityRelation

Die Klasse rrdf:IdentityRelation ist eine Subklasse von rrdf:Relation und ist die Klasse aller Beziehungen zwischen Erblassprädikaten, die in einer Identitätsbeziehung zueinander stehen (vgl. Kapitel 4.4.5).

rrdf:RoleRelation

Die Klasse rrdf:RoleRelation ist eine Subklasse von rrdf:Relation und ist die Klasse aller Beziehungen zwischen Erblassprädikaten, die in einer Rollenbeziehung zueinander stehen (vgl. Kapitel 4.4.5).

rrdf:CounterpartRelation

Die Klasse rrdf:CounterpartRelation ist eine Subklasse von rrdf:Relation und ist die Klasse aller Beziehungen zwischen Erblassprädikaten, die in einer Gegenpart-Beziehung zueinander stehen (vgl. Kapitel 4.4.5).

5.1.3 Rollenvokabular für Generalisierungsinformationen auf Schemaebene

Das Rollenvokabular für Generalisierungsinformationen auf Schemaebene ist eine Erweiterung des Rollenvokabulars für Generalisierungsinformationen auf Instanzebene (vgl. Kapitel 5.1.2). Anstatt der erblassenden und der erbenden Rolle, die mit rrdf:role

bzw. rrdf:player angegeben werden, wird hier die erblassende bzw. erbende Rollenklasse angegeben. Dadurch ist eine allgemeine Beschreibung der Generalisierungsvererbung möglich. Für Beispiele sei auf Kapitel 5.2.4 verwiesen. Für die Erklärung der Begrifflichkeiten sei auf Abbildung 25 verwiesen.

Um Generalisierungsinformationen auf Schemaebene erstellen zu können, wird folgendes Vokabular eingeführt:

rrdf:roleClass

Das Prädikat rrdf:roleClass ist eine Instanz von rdf:Property und gibt die Erblasserrollenklasse der Generalisierungsvererbung an.

Ein Tripple der Form:

```
_:n rrdf:roleClass C
```

besagt, dass C eine rrdf:RoleClass und zugleich die Erblasserrollenklasse ist. rdf:range von rrdf:roleClass ist rrdf:RoleClass.

rrdf:playerClass

Das Prädikat rrdf:playerClass ist eine Instanz von rdf:Property und gibt die erbende Rollenklasse der Generalisierungsvererbung an.

Ein Tripple der Form:

```
_:n rrdf:playerClass C
```

besagt, dass C eine rrdf:RoleClass und die erbende Rollenklasse ist. rdf:range von rrdf:playerClass ist rrdf:RoleClass.

Zur Verdeutlichung des Vokabulars und der Begrifflichkeiten wird in Abbildung 25 am Beispiel einer Rollenbeziehung eine Übersicht gegeben.

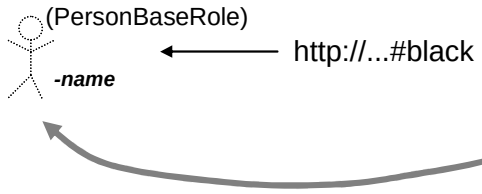
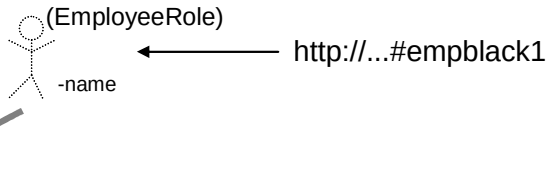
Superrolle	Subrolle
Erbe der Generalisierungsvererbung	Erblasser der Generalisierungsvererbung
erbende Rolle	Erblasserrolle
erbende Rollenklasse: PersonBaseRole	Erblasserrollenklasse: EmployeeRole
erbendes Prädikat: name	Erblasserprädikat: name
rrdf:playerProperty: http://...#name	rrdf:roleProperty: http://...#name
rrdf:player: http://...#black	rrdf:role: http://...#empblack1
rrdf:playerClass: http://...#PersonBaseRole	rrdf:roleClass: http://...#EmployeeRole
	

Abbildung 25: Übersicht der Begrifflichkeiten und des Vokabulars der Generalisierungsvererbung

5.2 Umsetzung der Anwendungsfälle im Web of Data

Auf Basis des zuvor definierten Vokabulars soll nun gezeigt werden, wie Rollen im Web of Data abgebildet werden können. Im Folgenden wird die Erstellung einer Rollenklassenhierarchie gezeigt, welche für die anschließende beispielhafte Umsetzung der vier Anwendungsfälle aus Kapitel 4.2.2 wieder verwendet wird. Die vier Anwendungsfälle beziehen sich jeweils auf das durchgängige Beispiel aus Kapitel 1.2 und stellen idealtypische Szenarien dar, mit denen der Informationsprovider im Web of Data konfrontiert wird. Die vollständige Umsetzung der Anwendungsfälle kann dem Anhang entnommen werden. Dadurch dass die Klassifizierung von Nicht-Informationsressourcen auf Entitätsebene keine Auswirkung auf die Rollenhierarchie bzw. auf Rollen im Allgemeinen hat, wird auf die Darstellung dieser verzichtet. Weiters sind folgende Namensraumangaben für Beispiele dargestellt in **Fehler! Verweisquelle konnte nicht gefunden werden.**, Abbildung 27, Abbildung 28, Abbildung 29 und Abbildung 30 gültig:

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.

@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.

@prefix owl: <http://www.w3.org/2002/07/owl#>.

@prefix foaf: <http://xmlns.com/foaf/0.1/>.

@prefix rrdf: <http://huemer.lstadler.net/role/rrdf.rdf#>.

5.2.1 Erstellung einer Rollenklassenhierarchie

Eine Rollenklassenhierarchie kann als ein kontextspezifisches Rollenvokabular betrachtet werden, das in diesem Fall zur Beschreibung verschiedener Rollen von Mr. Black verwendet wird. Mr. Black soll in einem allgemeinen Kontext als Person und in speziellen Kontexten als Angestellter, Student, Projektmanager und als Privatperson beschrieben werden. Diese Kontexte stellen zugleich auch die Rollenklassenbezeichnung dar. In Abbildung 26 wird die Erstellung einer Rollenklassenhierarchie in N3 Notation gezeigt und rechts daneben die Rollenklassenhierarchie als Baum dargestellt. Die Rollenklassen (Rollentypen) sind Instanzen von rrdf:RoleClass und werden mit dem Prädikat rrdf:roleSuperClass verlinkt. Durch diese Verlinkung entsteht die Rollenklassenhierarchie.

Als nächstes werden Rollentypen (Rollenklassen) mit qualifizierenden Attributen zu qualifizierten Rollen gemacht und zwar all jene Rollentypen, welche mehrmals von ein und derselben Entität eingenommen werden. Das qualifizierende Attribut wird mittels dem Prädikat owl:hasKey zum Rollentyp angegeben. Im Falle von Mr. Black ist das der Rollentyp EmployeeRole, welcher über das Attribut rh:employer und seine Rollenbeziehung qualifiziert wird. In Abbildung 26 ist die qualifizierte Rolle im Rollenhierarchiebaum unterstrichen dargestellt. (Anmerkung: Die Rollenbeziehung, also das Prädikat rrdf:roleOf sollte immer Teil des Schlüssels sein.)

File: http://huemer.lstadler.net/role/rh.rdf

```
@prefix rh: <http://huemer.lstadler.net/role/rh.rdf #>.
:                    :
rh:PersonBaseRole a rrdf:RoleClass.
rh:EmployeeRole a rrdf:RoleClass,
                    rrdf:roleSuperClass rh:PersonBaseRole;
                    owl:hasKey (rrdf:roleOf, rh:employer).
rh:ProjectManagerRole a rrdf:RoleClass;
                    rrdf:roleSuperClass rh:EmployeeRole.
rh:StudentRole a rrdf:RoleClass;
                    rrdf:roleSuperClass rh:PersonBaseRole.
rh:PrivatePersonRole a rrdf:RoleClass;
                    rrdf:roleSuperClass rh:PersonBaseRole.
```

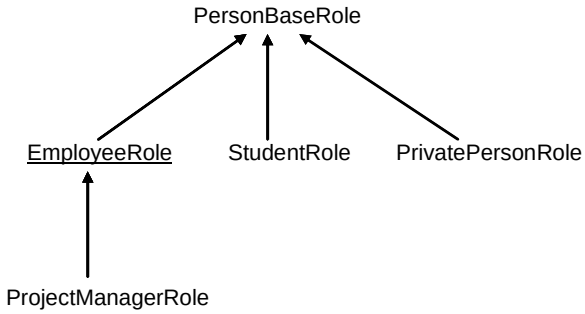


Abbildung 26: Repräsentation einer Rollenklassenhierarchie mit rrdf Vokabular (links) und als Baum (rechts)

5.2.2 Anwendungsfall 1: Neuerstellung einer Rollenbeschreibung

Dieser Anwendungsfall geht davon aus, dass eine komplette Neubeschreibung einer Nicht-Informationsressource erstellt wird, d.h. es werden keine URIs für Rollen verwendet, die bereits im Web of Data existieren (vgl. Kapitel 4.2.2.1).

Um Mr. Black in verschieden Rollen zu beschreiben, werden für ihn verschiedene URIs vergeben, je eine URI pro Rolle. Mit jeder URI wird eine Beschreibung von Mr. Black in der jeweiligen Rolle erstellt (Modell im Sinne von [Boo06]). Diese Rollen (Modelle) werden auf Rollenebene mit dem Attribut rrdf:roleType klassifiziert und mit dem Attribute rrdf:roleOf verknüpft, wodurch die in Abbildung 27 als Baum dargestellte Rollenhierarchie entsteht. Die hierarchisch übergeordnete Rolle ist der Player der hierarchisch untergeordneten Rolle. D.h. x:black ist der Player von x:empBlack1 und x:empBlack2. Innerhalb der Rollenhierarchie findet keine Vererbung, weder Attribut- noch Typvererbung, statt. Es gibt lediglich einen Gültigkeitsbereich für Attribute. Dieser erstreckt sich über einen Zweig im Baum. D.h. für x:black besitzen alle Aussagen mit dieser URI, inklusive aller Aussagen zu untergeordneten Rollen, Gültigkeit. Für x:empBlack1 wiederum besitzen nur die Aussagen mit dieser URI, inklusive der untergeordneten Rolle (x:prjBlack), Gültigkeit, alle anderen Aussagen mit anderen URIs nicht.

NEU: File: http://huemer.lstalder.net/role/uc1/foaf.rdf

```
@prefix rh: <http://huemer.lstadler.net/role/rh.rdf #>.
@prefix x: <http://huemer.lstadler.net/role/uc1/foaf.rdf#>.
:                    :
x:black rrdf:roleType rh:PersonBaseRole;
                     foaf:name „Peter Black“.
x:empBlack1 rrdf:roleType rh:EmployeeRole;
                     rh:employer <http://www.unixy.net/uni>;
                     rh:salary „1000“;
                     rrdf:roleOf x:black.
x:prjBlack rrdf:roleType rh:ProjectManagerRole;
                     rh:project „SemTech“;
                     rh:task „QA“;
                     rrdf:roleOf x:empBlack1.
x:empBlack2 rrdf:roleType rh:EmployeeRole;
                     rh:employer <http://www.CompZ.net/>;
                     rh:salary „1500“;
                     rrdf:roleOf x:black.
```

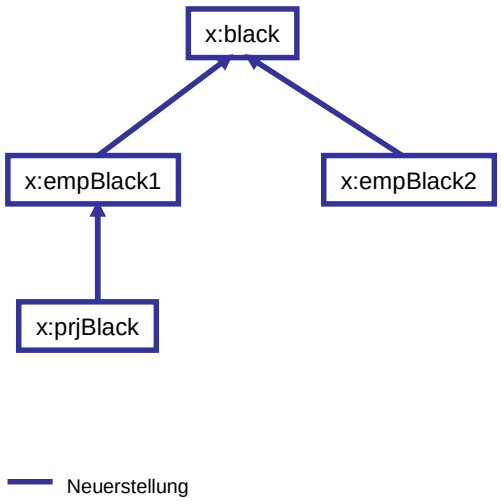


Abbildung 27: Anwendungsfall 1 – Neuerstellung von Rolleninstanzen (links) und hierarchische Darstellung dieser als Baum (rechts)

5.2.3 Anwendungsfall 2: Rollen als Spezialisierung

Diesen Anwendungsfall charakterisiert, dass bereits eine allgemeine Beschreibung von Mr. Black existiert (vgl. Kapitel 4.2.2.2). Die Unternehmen, für die Mr. Black arbeitet, möchten Informationen über ihn in der Rolle als Angestellten veröffentlichen. Sie haben allerdings keine Kontrolle über den Namensraum, in dem die allgemeine Beschreibung über Mr. Black erstellt worden ist. Sie besitzen allerdings Kenntnis von dieser Quelle und binden diese in ihre Beschreibung von Mr. Black mit ein.

Die Vorgehensweise bei der Erstellung der Beschreibung ist die gleiche wie in Anwendungsfall 1 – Top Down. In diesem Beispiel wird die Rollenklassenhierarchie, in Abbildung 26 dargestellt, verwendet. In Abbildung 28 sind Ressourcen, die neu erstellt werden, blau und dick gekennzeichnet. Bereits existierende Ressourcen über die der Ersteller keine Kontrolle hat, sind schwarz und dünn dargestellt. Bezüglich dem rdfs:seeAlso Link, in der bereits existierenden Ressource, möchte ich auf Kapitel 5.3 verweisen.

In Abbildung 28 wird Mr. Black in der jeweiligen Rolle von seinen Arbeitgebern eine URI gegeben, erstellt in einem Namensraum über den sie Kontrolle haben. Das Modell, das durch die Beschreibung mit der URI entsteht, wird auf Rollenebene mit rrdf:roleType klassifiziert. Auch das bereits existierende Modell im allgemeinen Kontext wird auf Rollenebene klassifiziert. Durch die Verlinkung mit dem Prädikat rrdf:roleOf wird die Rollenhierarchie erstellt (dargestellt in Abbildung 28 rechts oben).

File: http://huemer.lstadler.net/role/uc2/black.rdf

```
@prefix bl: <http://huemer.lstadler.net/role/uc2/black.rdf#>.
:                    :
bl:black a foaf:Person;
                     foaf:name „Peter Black“;
                     rdfs:seeAlso <http://huemer.lstadler.net/role/uc2/emp1.rdf>,
<http://huemer.lstadler.net/role/uc2/emp2.rdf>.
                     :
```

NEU: File: http://huemer.lstadler.net/role/uc2/emp1.rdf

```
@prefix un: <http://huemer.lstadler.net/role/uc2/emp1.rdf#>.
@prefix bl: <http://huemer.lstadler.net/role/uc2/black.rdf#>.
@prefix rh: <http://huemer.lstadler.net/role/rh.rdf#>.
:                    :
bl:black rrdf:roleType rh:PersonBaseRole.
un:empBlack1 rrdf:roleType rh:EmployeeRole;
                     rh:employer <http://www.unixy.net/uni>;
                     rh:salary „1000“;
                     rrdf:roleOf bl:black.
un:prjBlack rrdf:roleType rh:ProjectManagerRole;
                     rh:project „SemTech“;
                     rh:task „QA“;
                     rrdf:roleOf un:empBlack1.
```

NEU: File: http://huemer.lstadler.net/role/uc2/emp2.rdf

```
@prefix cz: <http://huemer.lstadler.net/role/uc2/emp2.rdf#>.
@prefix bl: <http://huemer.lstadler.net/role/uc2/black.rdf#>.
@prefix rh: <http://huemer.lstadler.net/role/rh.rdf#>.
:                    :
bl:black rrdf:roleType rh:PersonBaseRole.
cz:empBlack2 rrdf:roleType rh:EmployeeRole;
                     rh:employer <http://www.CompZ.net/>;
                     rh:salary „1500“;
                     rrdf:roleOf bl:black.
```

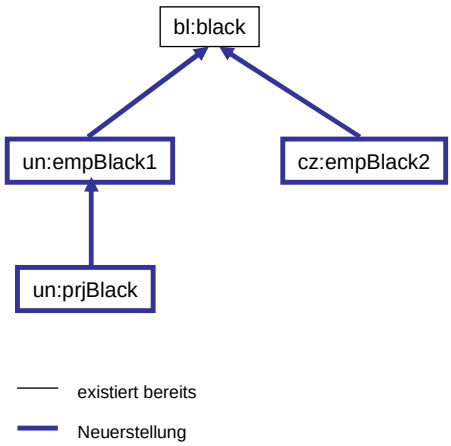


Abbildung 28: Anwendungsfall 2 – Rolleninstanzen als Spezialisierung in N3 Syntax (links) und als Baum (rechts)

5.2.4 Anwendungsfall 3: Rollen als Generalisierung

Der Anwendungsfall 3 beschreibt eine Bottom-Up Vorgehensweise, eine Situation, in der bereits Beschreibungen (Modelle) von Mr. Black existieren (vgl. Kapitel 4.2.2.3). In diesem Fall haben also die UniXY und die CompZ bereits Beschreibungen von Mr. Black als Angestellten und Projektmanager bzw. nur als Angestellten veröffentlicht. Mr. Black möchte auf Basis dieser Beschreibungen eine allgemeine Beschreibung über sich selbst erstellen und mit Generalisierungsinformationen versehen.

Diese neu zu erstellenden Teile sind in Abbildung 29 blau und dick gekennzeichnet. Die bereits existierenden Teile sind schwarz und dünn dargestellt. Bezüglich dem rdfs:seeAlso Link in der bereits existierenden Ressource möchte ich auf Kapitel 5.3 verweisen. Die

Arbeitgeber von Mr. Black haben kontextspezifische Beschreibungen über ihn, ohne die Umsetzung von Rollenbeziehungen im Web of Data veröffentlicht. Mr. Black ist in Kenntnis dieser Ressourcen gekommen und beschließt nun eine allgemeine Beschreibung von sich selbst anzufertigen und die bereits bestehenden Ressourcen als Rollen zu verlinken. Dabei möchte dieser die Aufwärtsvererbung von Prädikaten beschreiben. In diesem Beispiel wird erneut die Rollenklassenhierarchie, in Abbildung 26 dargestellt, verwendet.

Zuerst gibt sich Mr. Black selbst eine URI (bl:black), die in einem Namensraum definiert ist, über den er die Kontrolle hat und beschreibt sich im allgemeinen Kontext selbst. Anschließend werden die Modelle von Mr. Black mit dem Prädikat rrdf:roleType auf Rollenebene klassifiziert und mit dem Prädikat rrdf:roleOf miteinander verlinkt. Dadurch entsteht die in Abbildung 29 dargestellte Rollenhierarchie.
Im Anschluss wird die Aufwärtsvererbung von Prädikaten beschrieben. Mr. Black möchte von den Angestelltenrollen das Gehalt und den Arbeitgeber aufwärts zu der neu erstellten Rolle im allgemeinen Kontext vererben. Das Gehalt soll allerdings nur von jenen Angestelltenrollen aufsummiert werden, von denen er Kenntnis besitzt, während er alle Arbeitgeber auflisten möchte, die behaupten von ihm Arbeitgeber zu sein. D.h. einmal soll die Vererbung auf Instanzebene definiert werden und einmal auf Schemaebene. Die Beschreibung der Aufwärtsvererbung erfolgt durch einen leeren Knoten (Blanknode), ähnlich wie bei der Reification in [MaMi04a].

Die Beschreibung der Aufwärtsvererbung auf Instanzebene soll festhalten, dass das Gehalt von den Rollen un:empBlack1 und cz:empBlack2 zu bl:black aufwärts vererbt wird. Mit dem Prädikat rrdf:player wird der Erbe, also die übergeordnete Rolle angegeben. In diesem Fall die URI die Mr. Black als PersonBaseRole beschreibt. Mit dem Prädikat rrdf:playerProperty wird das erbende Prädikat angegeben. Besitzt das Prädikat bei der erbenden Rolle (bl:black) bereits eine Ausprägung, so wird dieses um die Ausprägungen der Erblasserprädikate ergänzt und nicht überschrieben. Mit rrdf:role werden die Erblasserrollen angegeben (un:empBlack1 und cz:empBlack2). Dies sind die Rollen, die Prädikate aufwärts vererben. Die Erblasserprädikate werden mit dem Prädikat rrdf:roleProperty angegeben, in diesem Beispiel das Prädikat rh:salary. Falls es eine Ausprägung besitzt, wird diese von den Rollen un:empBlack1 und cz:empBlack2 zur übergeordneten Rolle bl:black in das Prädikat rh:salary vererbt. Die Art der

Aufwärtsvererbung, d.h. in welcher Beziehung die Erblasserprädikate zueinander stehen, wird mit dem Prädikat rrdf:propertyRelation angegeben. Die Gehälter stehen in einer Rollenbeziehung zueinander (vgl. Kapitel 5.1.2).

File: http://huemer.lstadler.net/role/uc3/emp1.rdf

```
@prefix un: <http://huemer.lstadler.net/role/uc3/emp1.rdf#>.
@prefix rh: <http://huemer.lstadler.net/role/rh.rdf#>.
:                   :
un:empBlack1 un:employer <http://www.unixy.net/uni>;
                    rh:salary „1000“;
                    rdfs:seeAlso <http://huemer.lstadler.net/role/uc3/black.rdf>.
```

File: http://huemer.lstadler.net/role/uc3/emp2.rdf

```
@prefix cz: <http://huemer.lstadler.net/role/uc3/emp2.rdf#>.
@prefix ex: <http://huemer.lstadler.net/role/rh.rdf#>.
:                   :
cz:empBlack2 rh:employer <http://www.CompZ.net/>;
                    rh:salary „1500“;
                    rdfs:seeAlso <http://huemer.lstadler.net/role/uc3/black.rdf>.
```

NEU: File: http://huemer.lstadler.net/role/uc3/black.rdf

```
@prefix bl: <http://huemer.lstadler.net/role/uc3/black.rdf#>.
@prefix un: <http://huemer.lstadler.net/role/uc3/emp1.rdf/#>.
@prefix cz: <http://huemer.lstadler.net/role/uc3/emp2.rdf#>.
@prefix rh: <http://huemer.lstadler.net/role/rh.rdf#>.
:                   :
bl:black rrdf:roleType rh:PersonBaseRole;
                    foaf:name „Peter Black“;
un:empBlack1 rrdf:roleType rh:EmployeeRole;
                    rrdf:roleOf bl:black.
un:prjBlack rrdf:roleType rh:ProjectManagerRole;
                    rrdf:roleOf un:empBlack1.
cz:empBlack2 rrdf:roleType rh:EmployeeRole;
                    rrdf:roleOf bl:black.
[                   rrdf:player bl:black;
                    rrdf:playerProperty rh:salary;
                    rrdf:role un:empBlack1, cz:empBlack2;
                    rrdf:roleProperty rh:salary;
                    rrdf:propertyRelation rrdf:RoleRelation;
]
[                   rrdf:playerClass rh:PersonBaseRole;
                    rrdf:playerProperty rh:employer;
                    rrdf:roleClass rh:EmployeeRole;
                    rrdf:roleProperty rh:employer, un:employer;
                    rrdf:propertyRelation rrdf:RoleRelation;
]
```

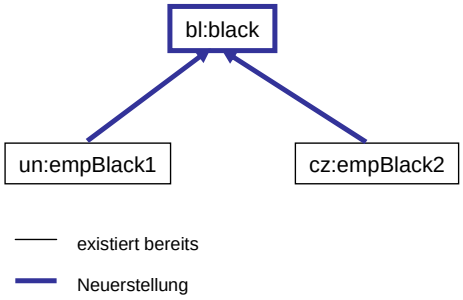


Abbildung 29: Anwendungsfall 3 – Rolleninstanzen als Generalisierung in N3 Syntax (links) und als Baum (rechts)

Bei der zweiten Beschreibung der Aufwärtsvererbung definiert Mr. Black, dass jede Rolle die auf Rollenebene als rh:PersonBaseRole klassifiziert ist von einer spezielleren Rolle

derselben Rollenhierarchie, die auf Rollenebene als rh:EmployeeRole klassifiziert ist, die Ausprägungen der Prädikate rh:employer und un:employer als Prädikat rh:employer vererbt bekommt. Mit dem Prädikat rrdf:playerClass wird die erbende Rollenklasse (rh:PersonBaseRole) angegeben. Mit dem Prädikat rrdf:roleClass wird die Erblasserrollenklasse (rh:EmployeeRole) angegeben. Als Erblasserprädikate, die mit rrdf:roleProperty definiert werden, sind hier zwei verschiedene Prädikate angegeben, nämlich rh:employer und un:employer. D.h. alle Ausprägungen dieser Attribute werden aufwärts vererbt in das Prädikat rh:employer, das mit dem Prädikat rrdf:playerProperty angegeben wird. Als Beziehungstyp zwischen den Erblasserprädikaten wird hier rrdf:RoleRelation angegeben (vgl. Kapitel 5.1.2).

5.2.5 Anwendungsfall 4: Anreicherung von vorhandenen Ressourcen

Der vierte Anwendungsfall beschreibt eine Situation, in der es bereits kontextspezifische Beschreibungen einer Nicht-Informationsressource gibt und diese verlinkt werden sollen (vgl. Kapitel 4.2.2.4).

Auch für diesen Anwendungsfall wird die in Abbildung 26 definierte Rollenklassenhierarchie verwendet. In Abbildung 30 sind alle neu zu erstellenden Komponenten blau und dick dargestellt, alle bereits existierenden Komponenten sind schwarz und dünn dargestellt. Bezüglich der rdfs:seeAlso Verweise in den bereits existierenden Ressourcen möchte ich auf Kapitel 5.3 verweisen.

Mr. Black möchte also Modelle, die ihn in einem bestimmten Kontext beschreiben, mit Rollenverlinkungen anreichern. Die Klassifizierung auf Rollenebene und die Rollenbeziehung zwischen Modellen (Rollen) soll in einer unabhängigen Informationsressource im Web of Data veröffentlicht werden. All jene, die Interesse an der Rollenverlinkung haben, können diese Informationsressource in der Applikation laden und erhalten eine Anreicherung der Daten über Mr. Black.
Bei dieser Anreicherung von bereits vorhandenen Ressourcen müssen die Modelle nur noch mit dem Prädikat rrdf:roleType auf Rollenebene klassifiziert werden und anschließend mit dem Prädikat rrdf:roleOf die Rollebeziehungen zwischen den Rollen beschrieben werden. Auf die Angabe von Generalisierungsinformationen wird in diesem Beispiel der Einfachheit halber verzichtet.

File: http://huemer.lstadler.net/role/uc4/black.rdf

```
@prefix bl: <http://huemer.lstadler.net/role/uc4/black.rdf#>.
:                    :
bl:black a foaf:Person;
                     foaf:name „Peter Black“;
                     rdfs:seeAlso <http://huemer.lstadler.net/role/uc4/myroles.rdf>.
                     :
```

File: http://huemer.lstadler.net/role/uc4/emp1.rdf

```
@prefix un: <http://huemer.lstadler.net/role/uc4/emp2.rdf#>.
@prefix rh: <http://huemer.lstadler.net/role/rh.rdf#>.
:                    :
un:empBlack1 un:employer <http://www.unixy.net/uni>;
                     rh:salary „1000“;
                     rdfs:seeAlso <http://huemer.lstadler.net/role/uc4/myroles.rdf>.
un:prjBlack rh:project „SemTech“;
                     rh:task „QA“;
                     rdfs:seeAlso <http://huemer.lstadler.net/role/uc4/myroles.rdf>.
```

File: http://huemer.lstadler.net/role/uc4/emp2.rdf

```
@prefix cz: <http://huemer.lstadler.net/role/uc4/emp2.rdf#>.
@prefix ex: <http://huemer.lstadler.net/role/rh.rdf#>.
:                    :
cz:empBlack2 rh:employer <http://www.CompZ.net/>;
                     rh:salary „1500“;
                     rdfs:seeAlso <http://huemer.lstadler.net/role/uc4/myroles.rdf>.
```

NEU: File: http://huemer.lstadler.net/role/uc4/myroles.rdf

```
@prefix rh: <http://huemer.lstadler.net/roles/rh.rdf#>.
@prefix un: <http://huemer.lstadler.net/role/uc4/emp1.rdf#>.
@prefix cz: <http://huemer.lstadler.net/role/uc4/emp2.rdf#>.
@prefix bl: <http://huemer.lstadler.net/role/uc4/black.rdf#>.
:                    :
bl:black rrdf:roleType rh:PersonBaseRole.
un:empBlack1 rrdf:roleType rh:EmployeeRole;
                     rrdf:roleOf bl:black.
un:prjBlack rrdf:roleType rh:ProjectManagerRole;
                     rrdf:roleOf un:empBlack1.
cz:empBlack2 rrdf:roleType rh:EmployeeRole;
                     rrdf:roleOf bl:black.
```

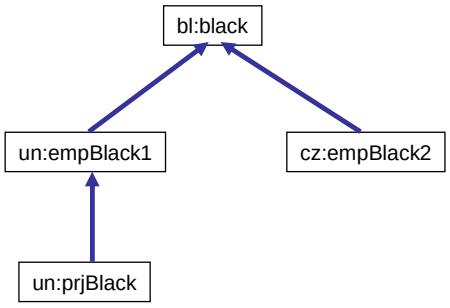


Abbildung 30: Anwendungsfall 4 – Anreicherung von Rolleninstanzen in N3 Syntax (links) und als Baum (rechts)

5.3 Maßnahmen zur Verbesserung der Navigierbarkeit

Mit Ausnahme von Anwendungsfall 1, wo alle Statements über Mr. Black in einer Informationsressource zusammengefasst sind, ist es nicht ohne weiteres möglich in beide Richtungen, nämlich von der Subrolle zur Superrolle und vice versa, zu navigieren (vgl. Kapitel 4.5.2). Um eine vollständige Navigierbarkeit zu erreichen, bedarf es der rdfs:seeAlso Verweise, welche in Abbildung 28, Abbildung 29 und Abbildung 30 schwarz und kursiv dargestellt sind.

Ein derartiger Verweis kann nicht nur als Beitrag zur Navigierbarkeit gesehen werden, sondern auch als ein Beitrag zur Evaluierung der Vertrauenswürdigkeit von Aussagen. Dabei kann davon ausgegangen werden, dass die Informationsressource, in welcher der rdfs:seeAlso Verweis steht, der Informationsressource auf die verwiesen wird, vertraut.

Teil C – Visualisierung und Browsing von Rollen mittels Tabulator

6. Der Linked Data Browser Tabulator 102
7. Der rollenfähige Tabulator 112

6. Der Linked Data Browser Tabulator

Im Folgenden wird auf den generischen Linked Data Browser Tabulator eingegangen und ausgewählte Aspekte der Architektur und Abläufe von bzw. in Tabulator dargestellt. Weiters wird auf die Benutzerschnittstelle von Tabulator eingegangen.

6.1 Einleitung

Tabulator ist ein generischer RDF Browser, der es Benutzern möglich macht RDF Links über Ressourcengrenzen hinweg zu folgen und diese zu analysieren. Während ein Web Browser zwischen Dokumenten navigiert, navigiert ein Semantic Web Browser zwischen Beziehungen (Prädikaten) in einem Web von Daten. Nichts desto trotz ist für einen Semantic Web Browser die zugrunde liegende Dokumentenstruktur wichtig, beispielsweise wenn es um Vertrauensevaluierung von Statements geht oder wenn es darum geht eventuelle Netzwerkfehler für den Benutzer nachvollziehbar zu machen. [BCC+06]

Tabulator ist im Rahmen dieser Diplomarbeit dahingehend adaptiert worden, dass Rollen in einer, aufgrund ihrer Semantik, geeigneten Art und Weise dargestellt werden. Da dieser Browser einer der wenigen open source und content-unabhängigen Tools ist die RDF Daten über Dokumentgrenzen hinaus verfolgen können, ist Tabulator für diese Arbeit ausgewählt worden. Tabulator gilt, da seine Entwickler aus den Reihen der Initiatoren des Web of Data stammen, als Best Practice Beispiel was die Darstellung und die verwendeten Technologien bei der Implementierung betrifft. In der neuesten Version ist Tabulator nicht nur ein Linked Data Browser und Navigator, sondern auch ein Editor, womit RDF Daten geändert, hinzugefügt und gelöscht werden können. Auf letzteren Aspekt wird hier jedoch nicht näher eingegangen. [BCC+06] [BHL+07]

Nachdem vielfach kritisiert worden ist, dass es zu wenige Anwendungen gibt, die dem Informationsprovider im Web of Data den Nutzen von RDF Verlinkungen aufzeigen, was als Grund für die stagnierende Benutzerteilname im Web of Data genannt worden ist, hat Tim Berners-Lee persönlich im Herbst 2005 mit der Entwicklung von Tabulator begonnen. Im Laufe der Zeit ist Tabulator von seinen Studenten weiterentwickelt, optimiert und adaptiert worden. Das Ziel war, einen RDF Browser zu entwickeln, um bei neuen Benutzern Interesse am Web of Data zu wecken und um Entwicklern von RDF- Daten

einen Ansporn zu geben weitere Daten im Web of Data, unter Einhaltung von promoteten Standards, zu veröffentlichen und ihnen gleichzeitig die Möglichkeit zu geben nachzuvollziehen, wie ihre veröffentlichten Daten im Web of Data integriert sind. Ein RDF Browser ist quasi eine Belohnung für den Informationsprovider, welcher ein unmittelbares Feedback in Form der Datendarstellung erhält. [BCC+06] [Tab05]

6.2 Architektur von Tabulator

Die Anforderungen an Tabulator waren vielfältig, einerseits ist ein domain-unspezifisches Browsing gefordert worden und andererseits hat man die Möglichkeit schaffen wollen Daten domainspezifisch analysieren zu können. Die Implementierung von Tabulator soll ein Beispiel abgeben, wie Semantik Web Applikationen mit Web 2.0 Technologien umgesetzt werden können. Weiters soll die Benutzung von Tabulator so einfach wie möglich gemacht werden, d.h. der Installationsaufwand soll gleich Null betragen. Aus diesem Grund ist Tabulator als eine browserbasierte Javascript-Anwendung unter Verwendung von asynchronem Javascript entwickelt worden. Tabulator läuft lokal auf der Benutzermaschine und kommt ohne serverseitige Skripts aus. Es wäre auch möglich gewesen eine serverseitige Anwendung zu entwickeln. Dies würde jedoch dem dezentralen Grundgedanken des Web of Data widersprechen und auch keine Weiterentwicklung von wettbewerbsfähigen und austauschbaren Datenbrowsern zulassen. [BCC+06]

6.2.1 Asynchronous Javascript and RDF (AJAR)

Tabulator verwendet bei der Umsetzung asynchrones Javascript und soll als Beispielanwendung dienen, wie Web 2.0 Technologien mit Semantic Web Technologien verknüpft werden können. In Web 2.0 wird asynchrones Javascript in Verbindung mit XML als AJAX bezeichnet. XML bietet sich als Austauschformat zwischen Client und Server an, da es bereits viele XML Parser gibt, welche die Daten schnell verarbeiten können. In Tabulator ist eine AJAR (Asynchronous Javascript and RDF) Library erstellt worden. Diese bietet Datenabwicklungsfunktionen für asynchrone Semantic Web Applikationen, beispielsweise den Aufbau einer Webverbindung oder die Verarbeitung und Speicherung der RDF-Tripple in einer internen Datenstruktur (vgl. Kapitel 6.2.2). [BCC+06] [BHL+07] [Tab05]

Wird vom Benutzer eine URI angefragt, welche noch nicht im internen Datenstore (vgl. 6.2.2) von Tabulator ist, so wird versucht diese zu dereferenzieren. Bei der Darstellung von Ressourcen werden ereignisspezifische Funktionen registriert, welche dargestellte

Elemente aktualisieren. Diese Callbackfunctions werden abgefeuert, sobald sich der Status eines Elementes ändert. Das von Tabulator unterstützte Serialisierungsformat für RDF Graphen ist XML. Daher wird für die Client-Server-Kommunikation auch dieses Format verwendet. Soll nun eine URI dereferenziert werden, so schickt der Client an den jeweiligen Server eine Anfrage. Der Client arbeitet nach Absenden der Abfrage asynchron weiter. Sobald eine Antwort vom Server kommt, werden die Daten geparsed und in den internen Datenstore gespeichert. Nach der Vollendung dieses Vorgangs werden die dem Status (done, fail, …) entsprechenden Callbackfunctions abgefeuert, welche eine Aktualisierung der View vornehmen. Die jeweilige Funktion muss dabei selbst entscheiden, ob sie von der Änderung betroffen ist oder nicht. (vgl. Abbildung 31)

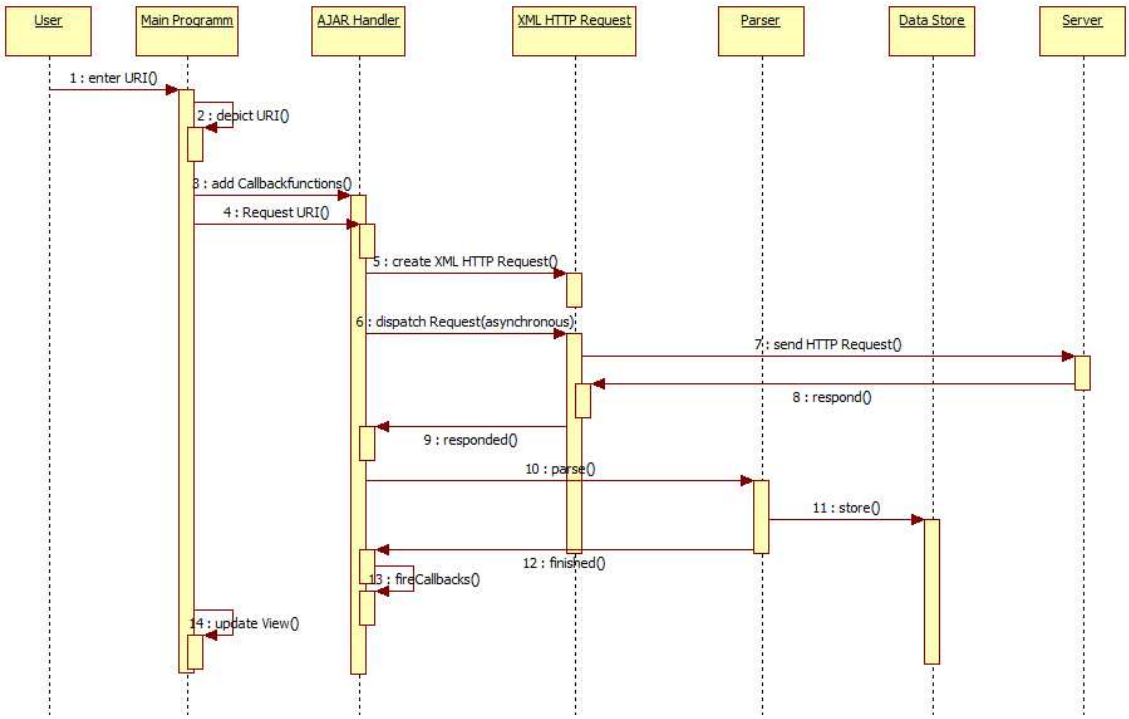


Abbildung 31: Sequenzdiagramm AJAR

6.2.2 Interner RDF-Quadruple-Store

Wird vom Benutzer eine URI im Tabulator abgefragt, so versucht Tabulator die URI zu dereferenzieren. Ist das Programm dabei erfolgreich, so wird die gesamte Informationsressource geladen und im Arbeitsspeicher gehalten. Die Daten werden in einem dreidimensionalen Array gespeichert (siehe Abbildung 32). Die erste Dimension des Arrays hat einen numerischen Index zwischen 0 und 3, wobei 0 für das Subjekt, 1 für das

Prädikat, 2 für das Objekt und 3 für die Herkunft eines RDF-Statements steht. Die zweite Dimension hat einen assoziativen Index, welcher die URI bzw. eine ID einer RDF-Ressource ist. Die dritte Dimension hat einen numerischen Index und ist ein Array von RDF-Statements. Ein RDF-Statement besteht aus einem Subjekt, Prädikat, Objekt und der Herkunft des Statements.

In der 2. Dimension ist jedes RDF Statement genau einmal in jedem Array indiziert. Beispielsweise das Statement

bl:black rrdf:roleType rh:PersonBaseRole

welches in der Informationsressource <http://huemer.lstadler.net/role/uc3/emp1.rdf> getätigt wird, kann auf folgende Weise wieder in der Datenstruktur, dargestellt in Abbildung 32, aufgefunden werden:

index[0][bl:black][*0*]

index[1][rrdf:roleType][*0*]

index[2][rh:PersonBaseRole][*0*]

index[3][<http://huemer.lstadler.net/role/uc3/emp1.rdf>][*0*]

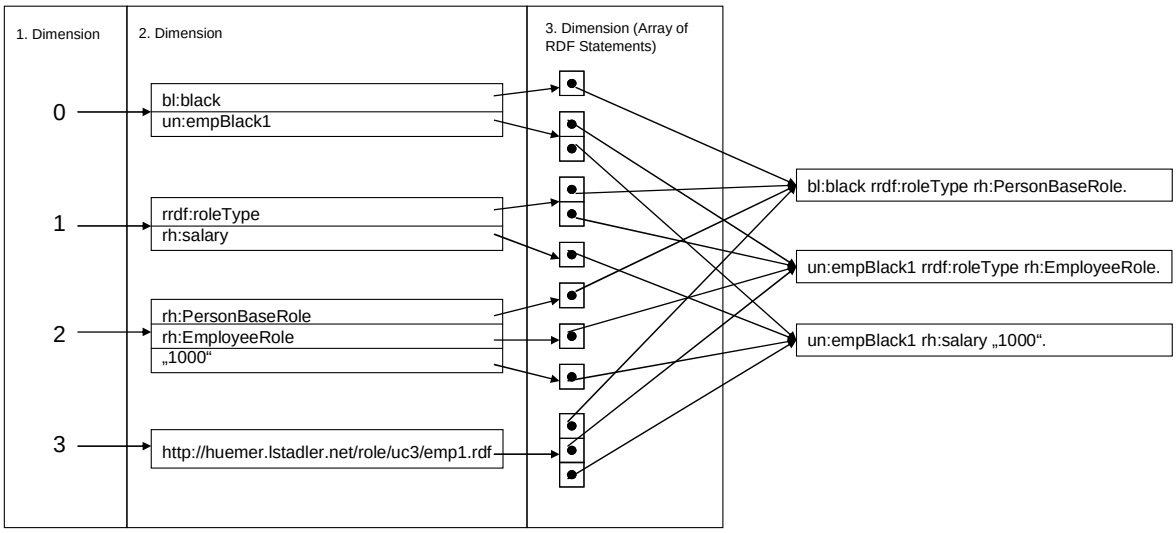


Abbildung 32: Datenstruktur in Tabulator

6.3 Benutzerschnittstelle

Die Benutzerschnittstelle ist, wie bereits erwähnt, ein HTML-Dokument, welches im Web Browser (derzeit wird nur Firefox unterstützt) dargestellt wird. Die Benutzerschnittstelle ist sehr intuitiv gestaltet und gliedert sich in zwei ineinander verzahnte Bereiche, in den Erkundungsmodus und in den Analysemodus. Im Erkundungsmodus werden Daten in einer Baumform dargestellt und können durch deren Verlinkung über Dokumentengrenzen

hinaus erkundet werden. Im Analysemodus, welcher sich unter dem Erkundungsmodus befindet, können mittels einer SPARQL Abfrage, Daten in verschiedenen Views analysiert werden.

6.3.1 Erkundungsmodus

Im Erkundungsmodus kann der Benutzer URIs eingeben, welche dereferenziert werden. Dem Benutzer wird der Erfolg der Anfrage in Form von färbigen runden Icons mitgeteilt. In Abbildung 33 ist beispielsweise die URI http://huemer.lstadler.net/role/praes/a.rdf#black angefragt worden. Tabulator hat Informationen zu der angefragten Nicht-Informationsressource gefunden und stellt diese dar. Diese URI ist das Subjekt. Alle Prädikate werden unter dieser mit dem dazugehörigen Objekt dargestellt. Es handelt sich um eine Person namens Peter Black. Diese Person kennt jemanden namens white und green. Weiters ist sie in der Nähe von bestimmten Koordinaten (Longitude und Latitude) ansässig.

Einerseits versucht Tabulator die Dokumentenstruktur vor dem Benutzer zu verbergen, andererseits ist diese notwendig um eventuelle Netzwerkfehler an den Benutzer zurück zu geben. Dies erfolgt mit runden färbigen Icons. Alle Ressourcen, die bereits im Tabulator geladen worden sind, d.h. alle, die bereits einmal erfolgreich dereferenziert worden sind, werden mit einem grünen Icon ausgezeichnet. Ressourcen, die bereits versucht worden sind zu laden, jedoch der Dereferenziervorgang gescheitert ist, werden mit einem roten Icon ausgezeichnet. Ressourcen die noch nicht angefragt worden sind, werden mit einem blauen Icon ausgezeichnet und jene, die sich in Abfrage befinden, werden mit einem gelben Icon dargestellt.

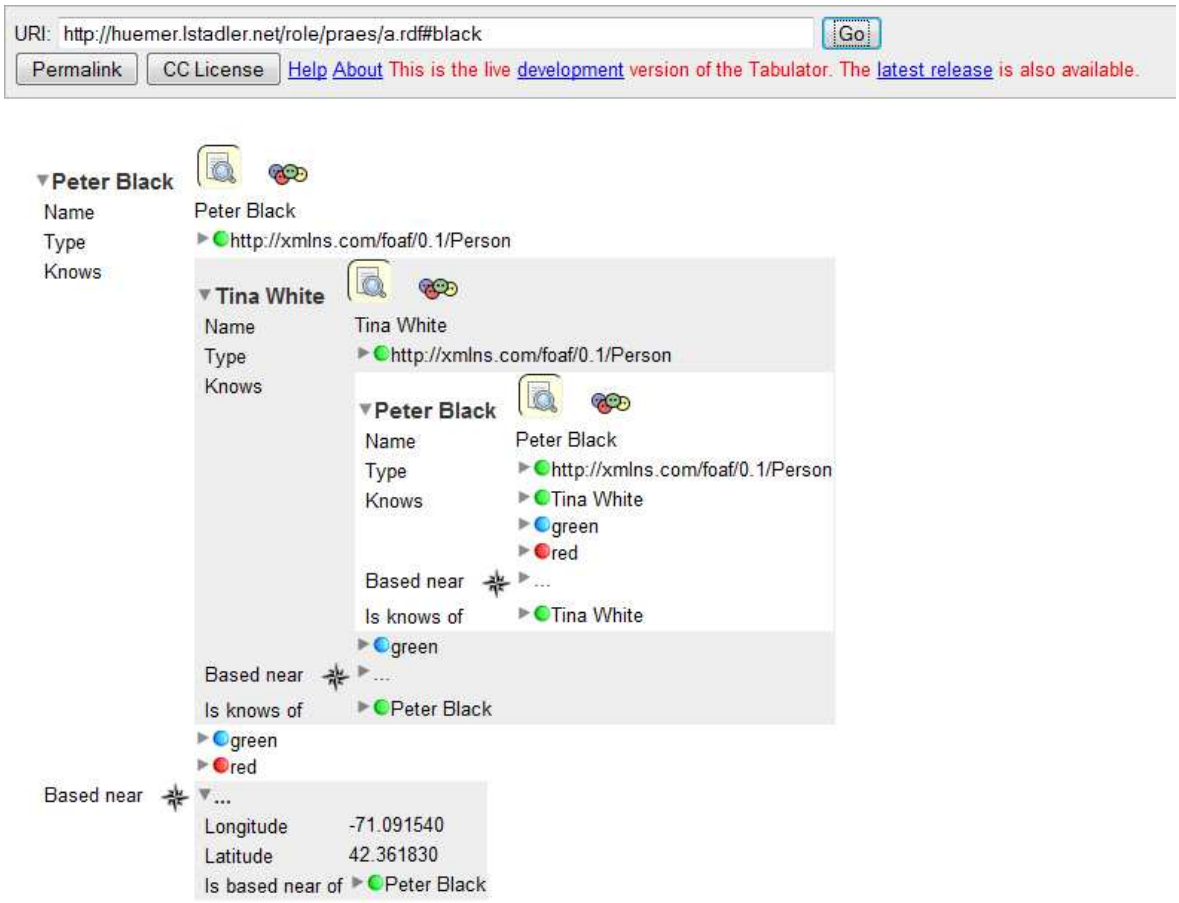


Abbildung 33: Tabulator: Erkundungsmodus

Der Benutzer kann dem RDF Graphen folgen, indem er auf die grauen Icons vor den färbigen klickt. Dadurch wird der Dereferenzierungsvorgang gestartet, sofern diese Ressource noch nicht dereferenziert worden ist. Ansonsten werden die entsprechenden Daten von der internen Datenstruktur dargestellt.

In der Darstellung des RDF-Graphen werden sowohl alle ausgehenden, als auch alle bekannten eingehenden (inversen) Verlinkungen angezeigt. [Tab05]

6.3.2 Analysemodus

Im Analysemodus können mit SPARQL Abfragen, die sich auf bereits geladenen Daten beziehen, getätigt werden. Die Ergebnisse von Abfragen können für die zur Analyse durch den Benutzer unterschiedlich dargestellt werden.

In Abbildung 34 wird eine Abfrage gestartet, welche alle Positionsdaten (Longitude und Latitude) zurückgibt. Die Abfrage erfolgt nicht nur auf die im Erkundungsmodus dargestellten Daten, sondern auf alle im internen Daten Store enthaltenen Daten.

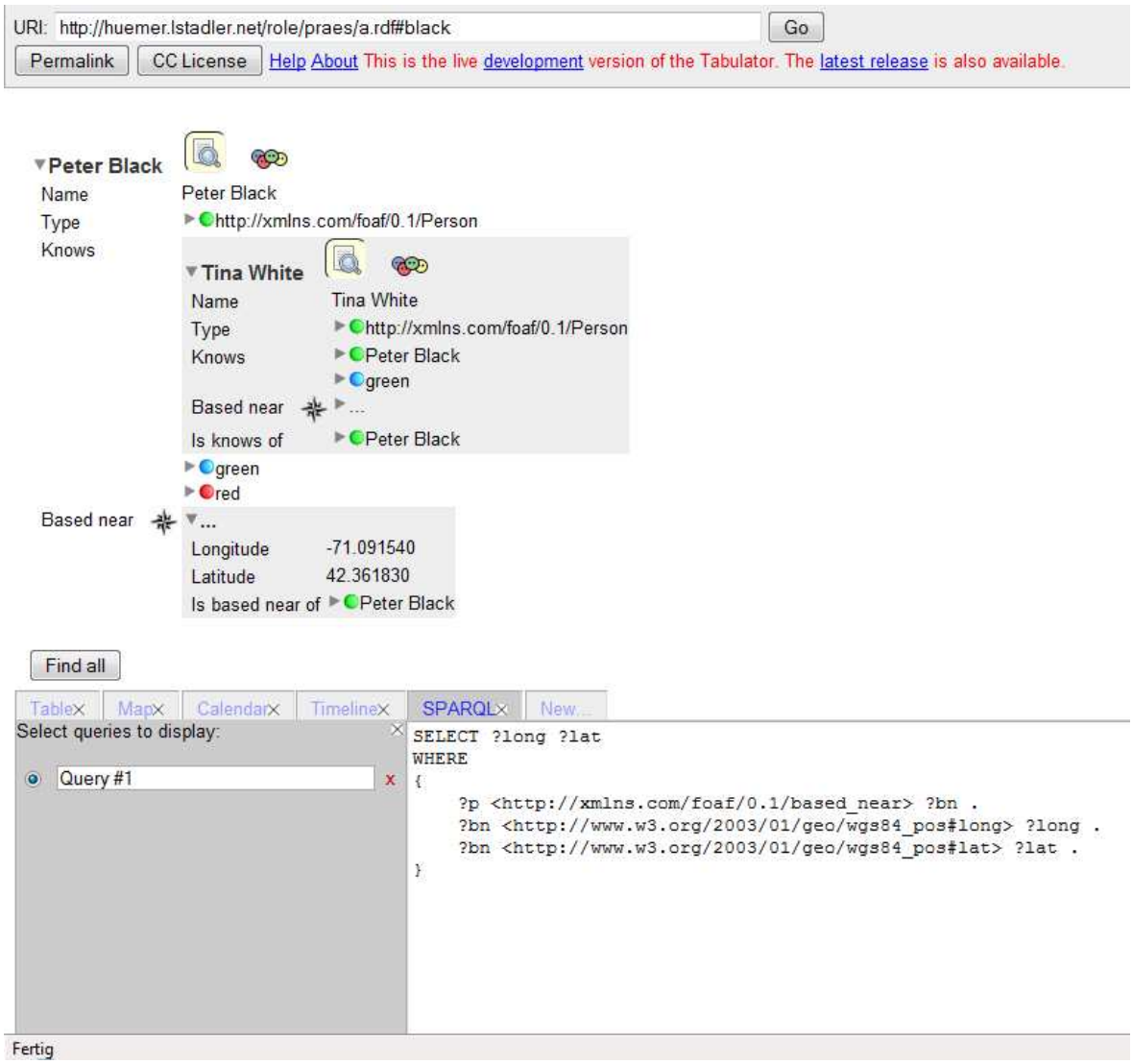


Abbildung 34: Analysemodus: SPARQL Abfrage

Das Ergebnis der Abfrage kann nun in der Table-View dargestellt werden (siehe Abbildung 35). Gibt es mehrere Abfragen, so kann zwischen diesen gewechselt werden. Nun, abhängig vom Ergebnis der Abfrage, kann dieses zur weiteren Analyse verwendet werden. In diesem Fall handelt es sich um ortsbezogene Koordinaten, welche in der Map-View dargestellt werden können.

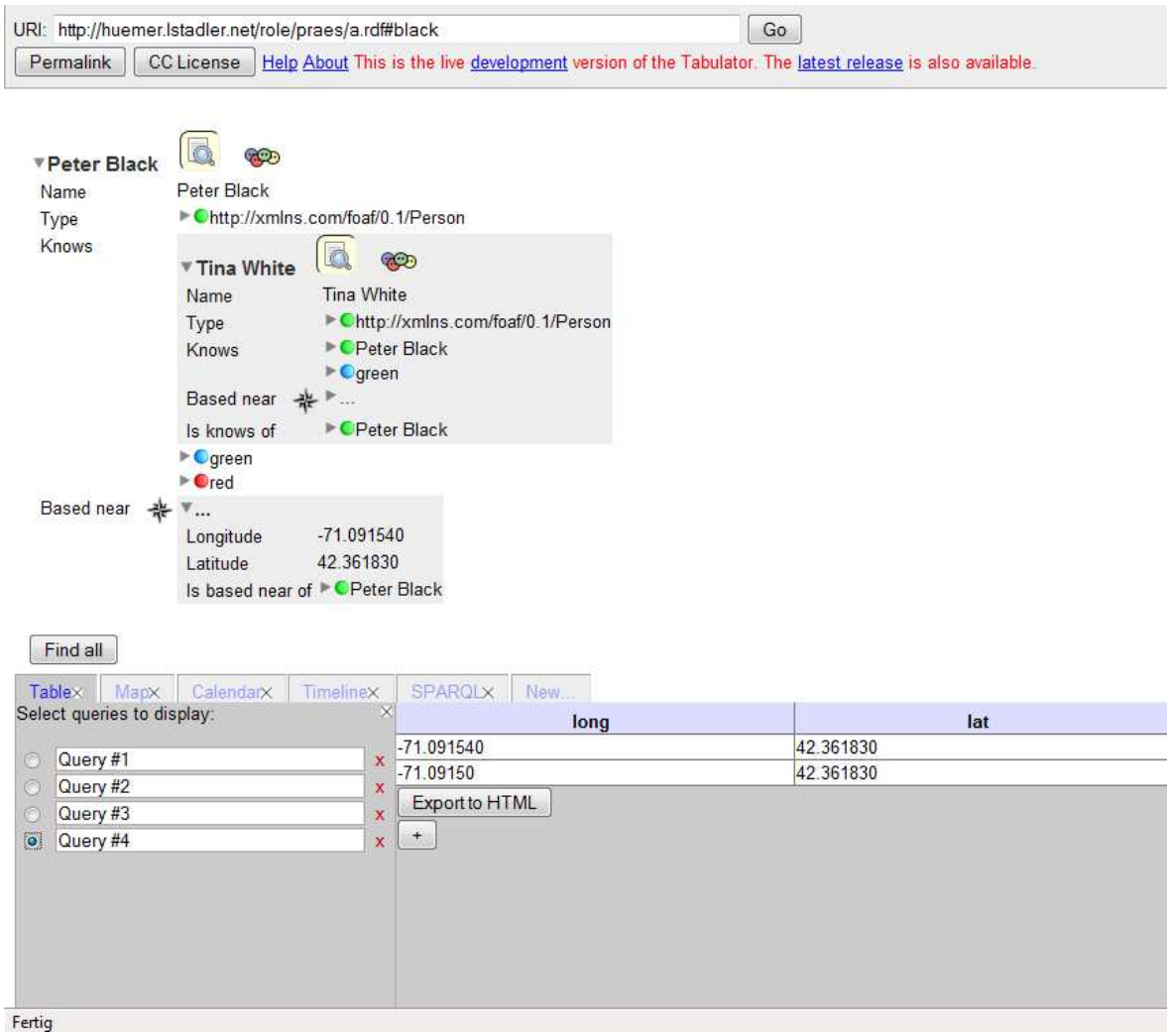


Abbildung 35: Analysemodus: Table-View

In Abbildung 36 werden die Koordinaten in der Map-View dargestellt. Koordinaten werden für den Benutzer aufbereitet und in der Karte eingezeichnet. Dabei können gleichzeitig mehrere Ergebnisse von Abfragen in einer Karte dargestellt werden.

Auf die weiteren Views, auf die Calendar-View und auf die Timeline-View, sowie auf die Möglichkeit Tabulator um neue Views zu erweitern, wird im Rahmen dieser Diplomarbeit nicht weiter eingegangen.

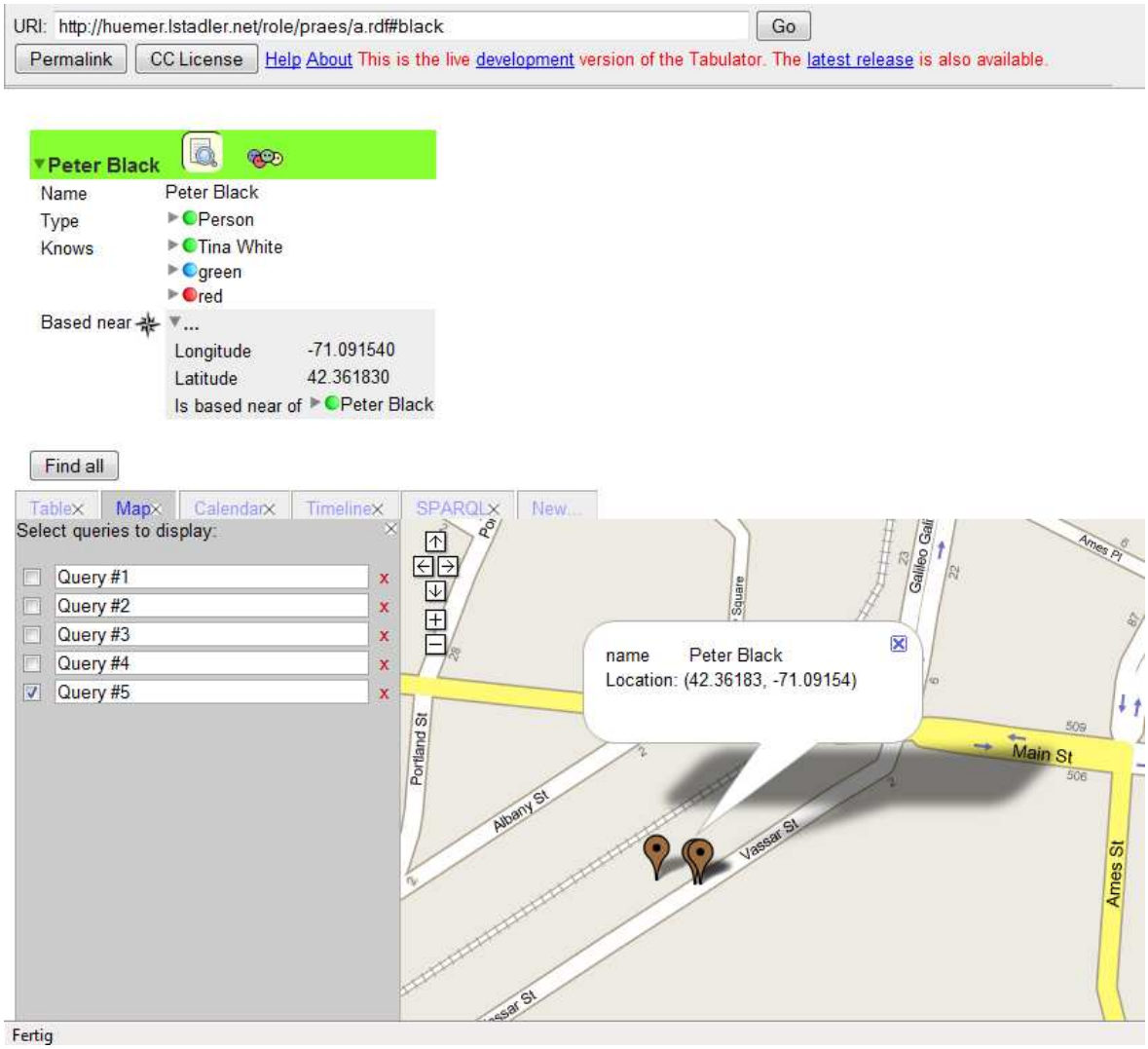


Abbildung 36: Analysemodus: Map-View

6.4 Ablauf einer Darstellung

In Abbildung 37 ist der Anwendungsfall, wenn ein Benutzer eine URI einer Nicht-Informationsressource zur Darstellung eingibt, in Form eines UML Aktivitätsdiagrammes dargestellt. Diese Abbildung soll einen kurzen Überblick über den internen Ablauf in Tabulator geben.

Zuerst wird die URI eingelesen um sie anschließend als Subjekt in Tabulator darzustellen. Ist die URI bereits einmal dereferenziert worden, so wird das Subjekt mit einem grünen Button versehen und die Darstellung um alle Prädikate und Objekte, mit der angefragten URI als Subjekt, ergänzt. Ist eines der darzustellenden Prädikate ein rdfs:seeAlso Verweis, so wird diesem gefolgt und die Objekt-URI gegebenenfalls dereferenziert und dargestellt.

Ist die vom Benutzer eingegebene URI noch nicht dereferenziert worden, so wird das Subjekt als Darstellung dieser URI als „angefragt“ mit einem gelben Button markiert.

Anschließend wird versucht die URI zu dereferenzieren. Ist der Versuch nicht erfolgreich, so wird das Subjekt entsprechend mit einem roten Button markiert. Ist der Versuch jedoch erfolgreich, so wird das vom Server erhaltene Dokument verarbeitet. Anschließend wird die Darstellung aktualisiert, d.h. grüne Buttons werden vergeben und die aktuelle Darstellung wird vervollständigt.

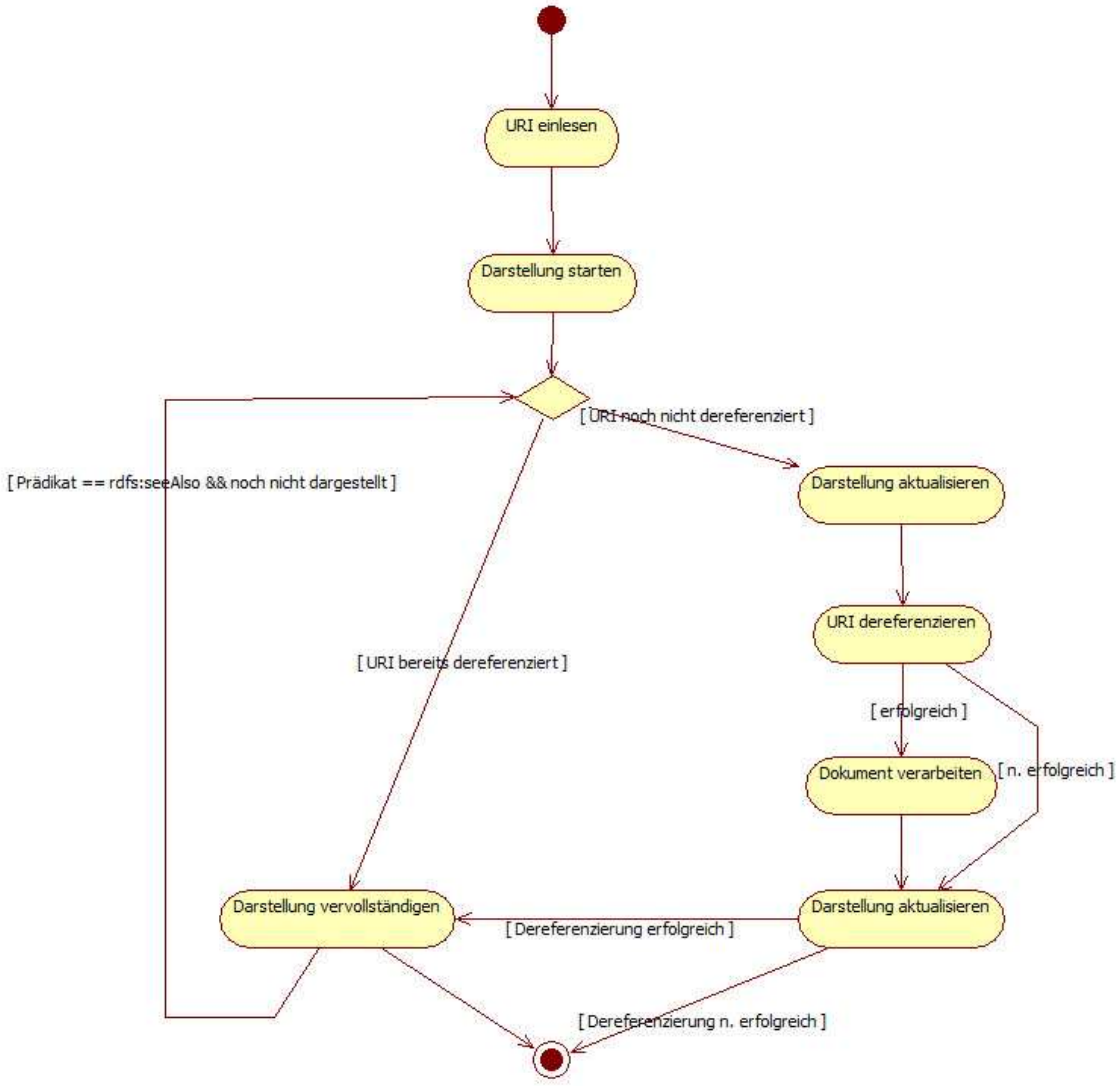


Abbildung 37: Aktivitätsdiagramm für den Anwendungsfall „Eingabe einer URI von einer NIR“

7. Der rollenfähige Tabulator

Ziel der Adaptierung von Tabulator ist, dass Rollen an ihre Semantik angepasst, dargestellt werden. D.h. es soll zum Ausdruck gebracht werden, dass einerseits Rollen entitätsspezifisch gebündelt werden und andererseits Rollen kontextspezifische Beschreibungen einer Nicht-Informationsressource sind. Dazu wird die von Tabulator verwendete Baumstruktur zur Darstellung von RDF-Graphen angepasst. Im Folgenden wird nun auf den rollenfähigen Tabulator eingegangen.

7.1 Adaptierung von Tabulator

Die Änderungen in Tabulator beziehen sich ausschließlich auf die Darstellungsebene und haben keine Auswirkung auf die in Tabulator verwendete Datenstruktur oder Programmlogik. Die Interpretation von Generalisierungsinformationen für die Aufwärtsvererbung von Daten ist in dieser Implementierung nicht umgesetzt worden. Es sind lediglich die dafür notwendigen Funktionen vorbereitet worden (mehr dazu siehe im Anhang).

Rollen werden im Tabulator wie ganz normale Ressourcen behandelt und dargestellt, mit drei Ausnahmen:

- Ist von einer Rolle die Superrolle bekannt, so kann direkt von der Subrolle zur Superrolle navigiert werden.
- Sind von einer Rolle Subrollen bekannt, so werden diese vor der Darstellung dereferenziert und hierarchisch unter der Superrolle dargestellt.
- Hat eine Rolle Subrollen, so können diese separat auf- bzw. zugeklappt werden.

Um diese Änderungen zu implementieren, muss die Aktivität „Darstellung vervollständigen“ aus dem Aktivitätsdiagramm in Abbildung 37 adaptiert werden. Die Aktivität „Darstellung vervollständigen“ wird in Abbildung 38 wiedergegeben. Zuerst wird eine Prädikatliste zum angefragten Subjekt generiert. Enthält die Prädikatliste Einträge, so wird diese so adaptiert, dass einerseits Prädikate die nicht dargestellt werden (beispielsweise rrdf:roleType) aus der Liste entfernt werden und andererseits die Prädikate sortiert werden. Nun beginnt die eigentliche Darstellung der Prädikate und Objekte bzw. der Rollen. Ist ein Prädikat ein Rollenprädikat, also rrdf:roleOf oder ein Subprädikat dieses, so ist dies wie eine Verlinkung zur Superrolle zu interpretieren und wird

dementsprechend dargestellt. Ist die gesamte Prädikatliste dargestellt worden, so wird eine neue generiert, nämlich jene mit den eingehenden Beziehungen (inversen Prädikaten). Diese Prädikatliste wird wiederum durch Adaption für die Darstellung vorbereitet und anschließend werden die Prädikate mit den dazugehörigen Objekten bzw. Rollen dargestellt. Enthält die Prädikatliste nun ein Rollenprädikat, so ist dies ein Verweis zu einer Subrolle und die rekursive Aktivität „Rolle darstellen“ wird gestartet. Befindet man sich am Ende der inversen Prädikatliste, so ist die Darstellung für das angefragte Subjekt abgeschlossen.

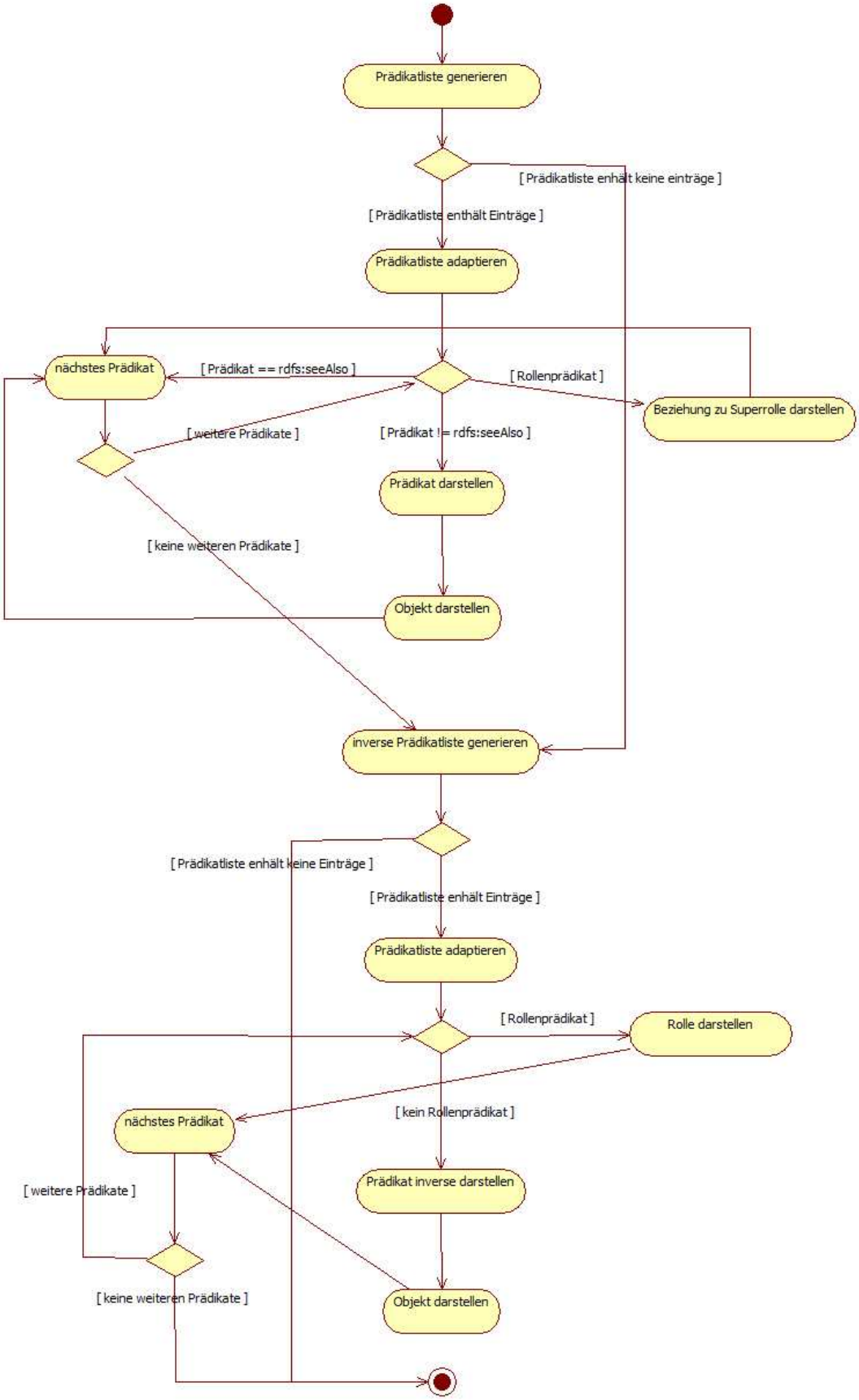


Abbildung 38: Aktivitätsdiagramm: Verfeinerung der Aktivität „Darstellung Vervollständigen“

7.2 Rollenfähige Benutzerschnittstelle

In Abbildung 39 wird das durchgängige Beispiel aus Kapitel 1.2 im rollenfähigen Tabulator dargestellt. Rollen sind mit einem kleinen färbigen dreieckigen Icon ausgezeichnet und deren Prädikate werden eingerückt dargestellt. Dieses Icon zeigt den Status der Rolle an. Ist es gelb gefärbt, so wird diese Rolle gerade dereferenziert und der dafür angefragte Server hat noch keine Antwort geliefert. Ist das Icon rot gefärbt, so ist eine Dereferenzierung versucht worden, diese hat allerdings fehlgeschlagen. Ein grün gefärbtes Icon zeigt einen erfolgreichen Dereferenziervorgang an. Weiters kann dieses Icon zum Auf- bzw. Zuklappen der Rolle verwendet werden.

Neben dem Rollenicon steht die Klassifizierung der Rolle auf Rollenebene. Darunter wird der Rollentyp abermals zum Ausdruck gebracht und stellt in einer inversen Beziehung zur Superrolle eine Verlinkung zu dieser dar. Darunter werden wiederum eventuelle qualifizierende Prädikate von qualifizierten Rollen kursiv ausgegeben. In weiterer Folge werden alle restlichen Prädikate dargestellt. Am Ende jeder Rollendarstellung werden eventuelle Subrollen ausgegeben.

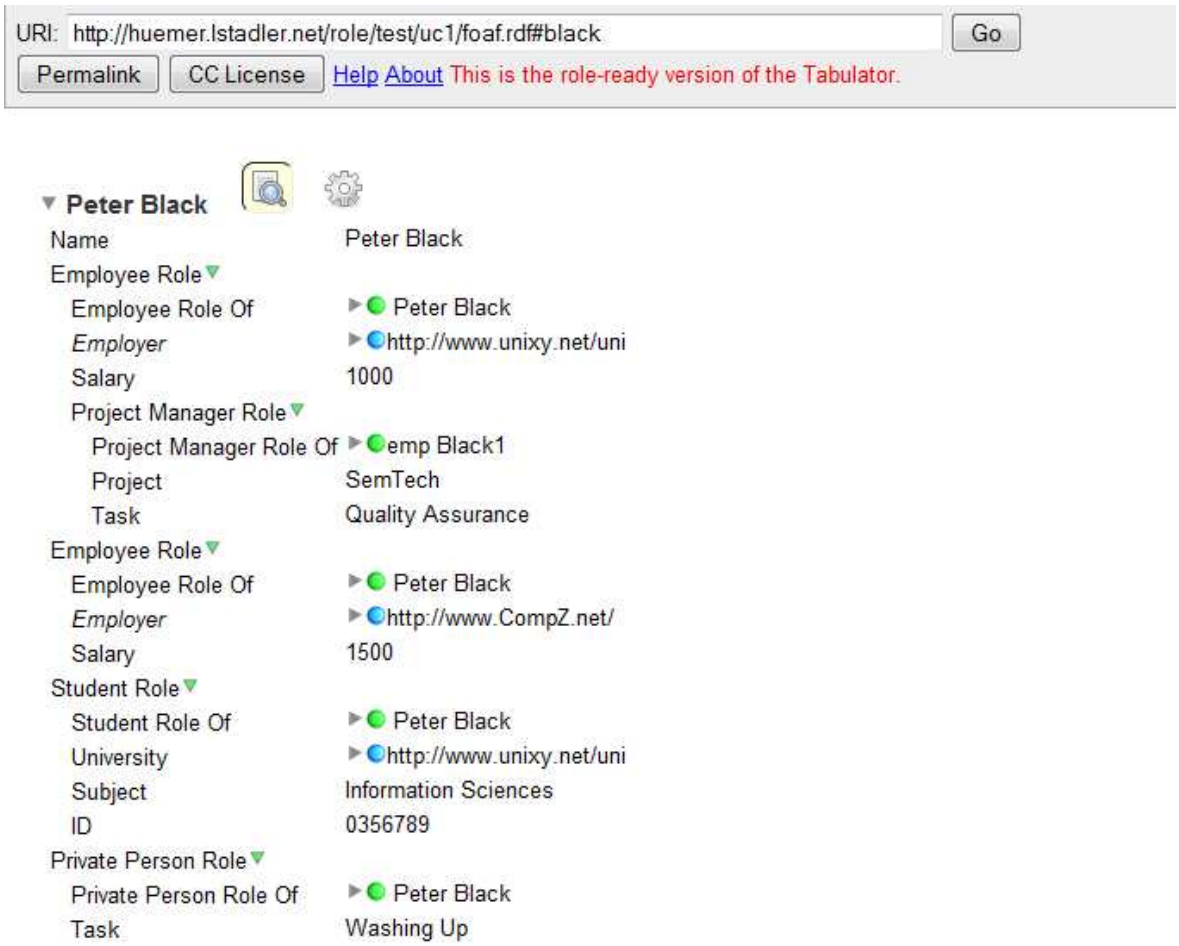


Abbildung 39: Role-ready Tabulator: Bsp. Mr. Black mit aufgeklappten Rollen

In Abbildung 40 werden alle Rollen zugeklappt gezeigt. Hier kann man sich einen Überblick darüber verschaffen, welche Rollen Peter Black einnimmt. Beim Aufklappen einer Rolle wird gleichzeitig auch der Inhalt aktualisiert, d.h. sind neue Aussagen über eine Rolle in die interne Datenstruktur von Tabulator hinzugefügt worden, so wird dies dabei berücksichtigt.

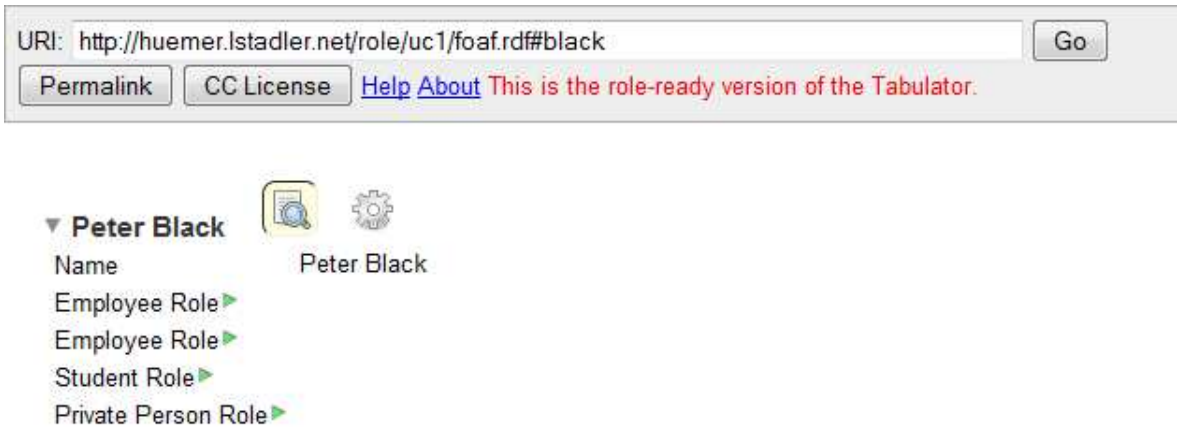


Abbildung 40: Role-ready Tabulator: Bsp. Mr. Black mit zugeklappten Rollen

8. Zusammenfassung

Diese Arbeit präsentiert ein RDF-basiertes Rollenmodell für das Web of Data. Dazu sind die Rahmenbedingungen im Web of Data im Vergleich zu den Rahmenbedingungen in objekt-orientierten Systemen analysiert worden.
Anschließend ist das Rollenmodell für das Web of Data auf Basis eines bestehenden objekt-orientiertes Rollenmodells [GSR96] entwickelt worden. Das Rollenmodell für das Web of Data unterstützt nicht nur den - in objekt-orientierten Systemen vorherrschenden - Top-Down Entwurf, sondern auch die - im Web of Data typische - Bottom-Up Vorgehensweise; insbesondere können Rollen mittels Aufwärtsvererbung generalisiert werden.

Das Rollenmodell für das Web of Data ist als RDF(S)-Vokabular umgesetzt worden. Die Anwendung dieses Rollenvokabulars ist beispielhaft anhand von Anwendungsfällen, welche typisch für Informationsprovider im Web of Data sind, gezeigt worden. Im Rahmen dieser Arbeit ist klar geworden, daß eine Umsetzung des Rollenmodells auf Basis von RDF(S) neben einem RDF(S)-Vokabular auch eine Erweiterung der RDF-Semantik im Hinblick auf Identität und Klassifizierung erforderlich macht: Ein URI identifiziert einerseits eine Entität (Resource) und andererseits - auf einer fein-granulareren Ebene - eine Rolle (ein kontextspezifisches Modell dieser Resource). Entsprechend wird auch unterschieden zwischen der Klassifizierung auf Entitätsebene und der Klassifizierung auf Rollenebene. In dieser Arbeit sind Identität und Klassifizierung auf diesen beiden Ebenen nur intuitiv beschrieben und an Beispielen gezeigt worden. Eine entsprechende Erweiterung der modell-theoretischen Semantik von RDF [Hay04] wird in zukünftigen Arbeiten behandelt.

Zum Ende der Arbeit ist der generische Linked Data Browser Tabulator vorgestellt und für das in dieser Arbeit vorgestellte Rollenmodell erweitert worden. Mit diesem rollenfähigen Data Browser können verteilte, um Rolleninformationen angereicherte RDF-Daten visualisiert und interaktiv erkundet werden. Der rollenfähige Tabulator wird zukünftig noch mit der Umsetzung von Aufwärtsvererbung vervollständigt, dafür sind in der vorliegenden Arbeit alle notwendigen vorbereitenden Maßnahmen bereits getroffen worden.

Diese Arbeit hat aufgezeigt, welch großen Mehrwert ein Rollenmodell für das Web of Data bringt. Es können damit Phänomene aus der Wirklichkeit realitätsgetreuer und intuitiver abgebildet werden. Rollenmodelle bieten eine Möglichkeit eine Entität mehrfach, in den jeweiligen Rollen (Kontexten), zu beschreiben und bündeln gleichzeitig die Beschreibungen dieser.

Vertreter des Ansatzes, dass es nur eine einzige Beschreibung pro Nicht-Informationsressource geben soll, können eine Rollenbeziehung als eine Äquivalenzbeziehung, also wie owl:sameAs, auffassen. All jene die eine kontextspezifische Beschreibung einer Nicht-Informationsressource wünschen können eine Rolle ihrer Semantik nach interpretieren - als eine kontextspezifische Beschreibung einer Entität mit enitätsspezifischer Bündelung.

Außerdem fördert ein Rollenmodell die Navigierbarkeit, indem es eine verteilte, autoritative Beschreibung von Nicht-Informationsressourcen unterstützt und liefert dadurch eine Alternative zu nicht-autoritativen Beschreibungen.

9. Literaturverzeichnis

[AnHa08] Antoniou, G., van Harmelen, F.: *A Semantic Web Primer*. Second Edition, The MIT Press, Cambridge, Massachusetts, London, England, 2008.

[BBZF+05] Burstein, M.H., Bussler, C., Zaremba, M., Finin, T.W., Huhns, M.N., Paolucci, M., Sheth, A.P., Williams, S.K.: *A Semantic Web Services Architecture.* In: IEEE Internet Computing, 9(5), 2005, pp. 72-81.

[BCC+06] Berners-Lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J., Lerer, A., Sheets, D.: *Tabulator: Exploring and Analyzing linked data on the Semantic Web.* Proceedings of the 3rd International Semantic Web User Interaction Workshop, 2006.

[BCH07] Bizer, C., Cyganiak, R., Heath, T.: *How to Publish Linked Data on the Web.* http://www4.wiwiss.fu-berlin.de/bizer/pub/LinkedDataTutorial/ , 2007. [letzter Besuch: 20.08.2009].

[BeBe08] Beckett, D., Berners-Lee, T.: *Turtle – Terse RDF Triple Language.* W3C Team Submission 14 January 2008. http://www.w3.org/TeamSubmission/turtle/ , 2008. [letzter Besuch: 20.08.2009].

[BePh08] Berrueta, D., Phipps, J.: *Best Practice Recipes for Publishing RDF Vocabularies.* W3C Working Group Note 28 August 2008. http://www.w3.org/TR/swbp-vocab-pub/ , 2008. [letzter Besuch: 02.07.2009].

[Ber98] Berners-Lee, T.: *Semantic Web Road map.* http://akira.ruc.dk/~jv/KIIS2004/roadmap.pdf , 1998. [letzter Besuch: 11.02.2009].

[Ber03] Berners-Lee, T.: W3C Mailing List Discussion Thread concerning: *Meaning of URIs in RDF documents.* http://lists.w3.org/Archives/Public/www-tag/2003Jul/0158.html , 2003. [letzter Besuch: 15.07.2009].

[Ber06a] Berners-Lee, T.: *Linked Data – Design Issues.* http://www.w3.org/DesignIssues/LinkedData.html , 2006. [letzter Besuch: 10.02 2009].

[Ber06b] Berners-Lee, T.: *An readable language for data on the Web – Notation 3.* http://www.w3.org/DesignIssues/Notation3 , 2006. [letzter Besuch: 24.09.2009].

[Ber07a] Berners-Lee, T.: *What do HTTP URIs Identify? – Design Issues.* http://www.w3.org/DesignIssues/HTTP-URI.html , 2007. [letzter Besuch: 12.08.2009].

[Ber07b] Berners-Lee, T.: Presentation about Linked Data. http://www.w3.org/2007/Talks/0130-sb-W3CTechSemWeb/#(24) , 2007. [letzter Besuch: 12.07.2009].

[Ber08] Berners-Lee, T.: Presentation about Linked Data. http://www.w3.org/2008/Talks/0617-lod-tbl/#(3) , 2008. [letzter Besuch: 17.07.2009].

[BFM05] Berners-Lee, T., Fielding, R., Masinter, L.: *Uniform Resource Identifier (URI): Generic Syntax.* IETF RFC 3986. http://www.ietf.org/rfc/rfc3986.txt , 2005. [letzter Besuch: 24.08.2009]

[BHL01] Berners-Lee, T., Hendler, J., Lassila, O.: *The Semantic Web.* In: Scientific American, 2001, pp. 34-43.

[BHL+07] Berners-Lee, T., Hollenbach, J., Lu, K., Presbrey, J., Pru d'ommeaux, E., Schraefel, M.C.: *Tabulator Redux: Writing Into the Semantic Web.* Technical Report. http://eprints.ecs.soton.ac.uk/14773/1/tabulatorWritingTechRep.pdf , 2007. [letzter Besuch: 12.07.2009].

[BKPP07] Boley, H., Kifer, M., Patranjan, P., Polleres, A.: *Rule Interchange on the Web.* In: Reasoning Web, Springer, 2007, pp. 269-309.

[BlPe06] Blumauer, A., Pelligrini, T.: *Semantic Web und semantische Technologien: Zentrale Begriffe und Unterscheidungen.* In: Semantic Web – Wege zur vernetzten Wissensgesellschaft, Pelligrini, T., Blumauer, A. (Eds), Springer, Berlin, Heidelberg, 2006, pp. 9-25.

[Boo06] Booth, D.: *URIs and the Myth of Resource Identity*. Proceedings of Identity, Reference, and the Web Workshop at the WWW Conference, 2006.

[BrMi07] Brickley, D., Miller, L.: *FOAF Vocabulary Specification 0.91*. http://xmlns.com/foaf/spec/ , 2007. [letzter Besuch: 20.08.2009].

[BSCT08] Bouquet, P., Stoermer, H., Cordioli D., Tummarello, G.: *An Entity Name System for Linking Semantic Web Data.* Proceedings of the 1st Workshop on Linked Data on the Web at WWW2008, Beijing, China, 2008.

[CBHS05] Carroll, J.J., Bizer, C., Hayes, P., Stickler, P.: *Named Graphs, Provenance and Trust.* Proceedings of the 14th International Conference on World Wide Web, WWW2005, Chiba, Japan, 2005.

[DeSc04] Dean, M., Schreiber G.: *OWL Web Ontology Language Reference.* W3C Recommendation 10 February 2004. http://www.w3.org/TR/owl-ref/#sameAs-def , 2004. [letzter Besuch: 20.08.2009].

[DMH+00] Decker, S., Merlnik, S., van Harmelen, F., Fensel, D., Klein, M., Broekstra, J., Erdmann, M., Horrocks, I.: *The Semantic Web: The Roles of XML and RDF*. In: IEEE Internet Computing 4(5), 2000, pp. 63-74.

[ESW09] ESW Wiki: *Equivalence Mining and Matching Frameworks*. http://esw.w3.org/topic/TaskForces/CommunityProjects/LinkingOpenData/EquivalenceMining , 2009. [letzter Besuch: 10.07.2009].

[GrBe04] Grant, J., Beckett, D.: *RDF Test Cases.* W3C Recommendation 10 February 2004. http://www.w3.org/TR/rdf-testcases/#ntriples , 2004. [letzter Besuch: 20.08.2009].

[GSR96] Gottlob, G., Schrefl, M., Röck, B.: *Extending Object-Oriented Systems with Roles.* ACM Transactions on Information Systems 14 (3), 1996, pp. 99-134.

[Gua92] Guarino, N.: *Concepts, Attributes, and Arbitrary Relations: Some Linguistic and Ontological Criteria for Structuring Knowledge Bases.* Data & Knowledge Engineering 8, 1992, pp. 249-261.

[Hal06] Halpin, H.: *Identity, Reference, and Meaning on the Web.* Proceedings of the Workshop on Identity, Meaning and the Web (IMW06) at WWW2006, Edinburgh, Scotland, 2006.

[Har05] Harris, S.: *SPARQL query processing with conventional relational database systems.* Proceedings of the International Workshop on Scalable Semantic Web Knowledge Base Systems, 2005.

[Hay04] Hayes, P.: *RDF Semantics*. W3C Recommendation 10 February 2004. http://www.w3.org/TR/rdf-mt/ , 2004. [letzter Besuch: 06.09.2009].

[Hea09] Heath, T.: *Linked Data? Web of Data? WTF?* http://tomheath.com/blog/2009/03/linked-data-web-of-data-semantic-web-wtf/ , 2009. [letzter Besuch: 10.07.2009].

[Hen09] Hendler, J.: *The Dark Side of the Semantic Web.* http://www.mindswap.org/blog/2006/12/13/the-dark-side-of-the-semantic-web/ , 2009. [letzter Besuch: 10.07.2009].

[HKRS08] Hitzler, P.; Krötzsch, M., Rudolph, S., Sure, Y.: *Semantic Web - Grundlagen.* 1. Auflage. Springer, Berlin, Heidelberg, 2008.

[HPPH05] Horrocks, I., Parsia, B., Patel-Schneider, P., Hendler, J.: *Semantic Web Architecture: Stack or Two Towers?* Proceedings of Principles and Practice of Semantic Web Reasoning, Third International Workshop, PPSWR 2005, Dagstuhl Castle, Germany, 2005.

[HTTP99] *HTTP/1.1.* HTTP Protocol. http://www.w3.org/Protocols/rfc2616/rfc2616.html, 1999. [letzter Besuch: 09.08.2009].

[JaWa04] Jacobs, I., Walsh, N.: *Architecture of the World Wide Web, Volume One.* W3C Recommendation 15 December 2004. http://www.w3.org/TR/2004/REC-webarch-20041215/ , 2004. [letzter Besuch: 10.07.2009].

[JGM07] Jaffri, A., Glaser, H., Millard, I.: *URI Identity Management for Semantic Web Data Integration and Linkage.* Proceedings of the Workshop on Scalable Semantic Web Systems, Vilamoura, Portugal, 2007.

[JGM08] Jaffri, A., Glaser, H., Millard, I.: *Managing URI Synonymity to Enable Consistent Reference on the Semantic Web.* Proceedings of the 1st IRSW2008 International Workshop on Identity and Reference on the Semantic Web, Tenerife, Spain, 2008.

[KBBF05] Kifer, M., de Bruijn, J., Boley, H., Fensel, D.: *A Realistic Architecture for the Semantic Web.* Proceedings of Rules and Rule Markup Languages for the Semantic Web, First International Conference, RuleML 2005, Galway, Ireland, 2005.

[KBM08] Kashyap, V., Bussler, C., Moran, M.: *The Semantic Web – Semantics for Data and Services on the Web.* Springer, Berlin, Heidelberg, 2008.

[KlCa04] Klyne, G., Carroll, J.J.: *Resource Description Framework (RDF): Concepts and Abstract Syntax.* W3C Recommendation 10 February 2004. http://www.w3.org/TR/rdf-concepts/ , 2004. [letzter Besuch: 24.01.2009].

[KPSD+04] Kagal, L., Paolucci, M., Srinivasan, N., Denker, G., Finin, T.W., Sycara, K.P.: *Authorization and Privacy for Semantic Web Services.* In: IEEE Intelligent Systems, 19(4), 2004, pp. 50-56.

[Kri95] Kristensen, B.B.: *Object-Oriented Modeling with Roles.* Proceedings of the International Conference on Object-Oriented Information Systems, Dublin, 1996.

[Lin09] *Linked Data – Connecting Distributed Data across the Web.* Homepage of the Linked Data Community. http://linkeddata.org. [letzter Besuch: 10.08.2009].

[MaMi04a] Manola, F., Miller, E.: *RDF Primer.* W3C Recommendation 10 February 2004. http://www.w3.org/TR/REC-rdf-syntax/ , 2004. [letzter Besuch: 21.12.2008].

[MaMi04b] Manola, F., Miller, E.: *RDF Vocabulary Description Language 1.0: RDF Schema.* W3C Recommendation 10 February 2004. http://www.w3.org/TR/rdf-schema/ , 2004. [letzter Besuch: 21.12.2008].

[Mer09] Merriam-Webster's Online Dictionary. http://www.merriam-webster.com/dictionary/role , 2009. [letzter Besuch: 15.07.2009].

[N3P00] Primer: Getting into RDF & Semantic Web using N3. http://www.w3.org/2000/10/swap/Primer.html , 2000. [letzter Besuch: 20.08.2009].

[NoRe04] Noy, N., Rector, A.: *Defining N-ary Relations on the Semantic Web: Use With Individuals.* W3C Working Draft 21 July 2004. http://www.w3.org/TR/2004/WD-swbp-n-aryRelations-20040721/ , 2004. [letzter Besuch: 23.02.2009].

[OAKS04] O'Hara, K., Alani, H., Kalfoglou, Y., Shadbolt, N.: *Trust Strategies for the Semantic Web.* ISWC 04 Workshop on Trust, Security and Reputation on the Semantic Web, 2004.

[ODG+07] Oren, E., Delbru, R., Gerke, S., Haller, M., Decker, S.: *ActiveRDF: Object-Oriented Semantic Web Programming.* Proceedings of the 16th International Conference on World Wide Web, WWW2007, Banff, Alberta, Canada, 2007.

[OHD08] Oren, E., Heitmann, B., Decker, S.: *ActiveRDF: embedding Semantic Web data into object-oriented languages.* In: Web Semantics: Science, Services and Agents on the World Wide Web, 6 (3), 2008, pp. 191-202.

[Oli07] Olivé, A.: *Conceptual modelling of Information Systems*. Springer, Berlin, Heidelberg, 2007.

[PAG06] Pérez, J., Arenas, M., Gutierrez, C.: *Semantics and Complexity of SPARQL.* Proceedings of the 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA, 2006.

[RAD03] Richardson, M., Agrawal, R., Domingos, P.: *Trust Management for the Semantic Web.* Proceedings of the Second International Semantic Web Conference, Sanibel Islands, Florida, 2003.

[ScNe88] Schrefl, M., Neuhold, E.J.: *Object class definition by generalization using upward inheritance.* Proceedings of the Fourth International Conference on Data Engineering, Los Angeles, California, USA, 1988.

[SCV07] Sauermann, L., Cyganiak, R., Völkel, M.: *Cool URIs for the Semantic Web.* DFKI Technical Memo TM-07-01. http://www.dfki.uni-kl.de/~sauermann/2006/11/cooluris/ , 2007. [letzter Besuch: 30.11.2008].

[SeGu06] Seemann, J., von Gudenberg, J.W.: *Software-Entwurf mit UML 2.* 2. Auflage, Springer, Berlin, Heidelberg, 2006.

[SHB06] Shadbolt, N., Hall, W., Berners-Lee, T.: *The Semantic Web Revisited.* IEEE Intelligent Systems 21(3), 2006, pp. 96-101.

[Ste00] Steinmann, F.: *On the representation of roles in object-oriented and conceptual modelling.* Data Knowledge Engineering 35(1), pp. 83-106, 2000.

[Tab05] *The Tabulator.* Homepage von und für Tabulator Entwickler. http://www.w3.org/2005/ajar/tab , 2005. [letzter Besuch: 12.08.2009].

[ToMa06] Tochtermann, K., Maurer, H.: *Semantic Web – Geschichte und Ausblick einer Vision.* Semantic Web – Wege zur vernetzten Wissensgesellschaft, Pelligrini, T., Blumauer, A. (Eds), Springer, Berlin, Heidelberg, 2006, pp. 1-6.

[Wan07] Wang, X.: *URI Identity and Web Architecture Revisited.* Technical Report. http://dfdf.inesc-id.pt/tr/web-arch#sec2 , 2007. [letzter Besuch: 13.07.2009].

[WaOl09] Wang, X., Oliveira A.L.: *Exploring HTTP Content Negotiation.* Proceedings of the 18th International Conference on World Wide Web, WWW2009, Madrid, Spain, 2009.

[WJS95] Wieringa, R.; de Jonge, W.; Spruit, P.: *Using Dynamic Classes and Role Classes to Model object Migration.* In: TAPOS 1(1), 1995, pp. 61-83.

[Zam05] Zambonini, D.: *Is Web 2.0 killing the Semantic Web?* O'Reilly XML Blog. http://www.oreillynet.com/xml/blog/2005/10/is_web_20_killing_the_semantic.html , 2005. [letzter Besuch: 17.07.2009].

[ZhZh08] Zhu, H., Zhou, M.: *Roles in Information Systems: A Survey*. In: IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews, 38(3), 2008.

10. Anhang

Der Anhang beinhaltet Details der Implementierung. In Unterkapitel A.1 wird das RDF-XML File mit der Definition des Rollenvokabulars präsentiert. Unterkapitel A.2 präsentiert die für die Implementierung der Anwendungsfälle definierte Rollenhierarchie. Die Unterkaptitel A.3 – A.6. präsentieren die vollständige Lösung der Anwendungsfälle von Kapitel 4.2.2. In Unterkapitel A.7 erfolgt eine Übersicht über die Änderungen die im Quellcode von Tabulator vorgenommen worden sind um Rollen ihrer Semantik entsprechend darzustellen. In Unterkapitel A.8 erfolgt eine Einführung in den Quellcode von Tabulator.

A.1 Rollenvokabular

File: http://huemer.lstadler.net/role/rrdf.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:rrdf="http://huemer.lstadler.net/role/rrdf.rdf#">

 <rdfs:Class rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#RoleClass">
        <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>RoleClass</rdfs:label>
        <rdfs:comment>This is the class of role-classes.</rdfs:comment>
 </rdfs:Class>

 <rrdf:RoleClass rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#Role">
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>Role</rdfs:label>
        <rdfs:comment></rdfs:comment>
 </rrdf:RoleClass>

 <rdf:Property rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#roleOf">
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>roleOf</rdfs:label>
        <rdfs:comment></rdfs:comment>
 </rdf:Property>

 <rdf:Property rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#specRoleOf">
        <rdfs:subPropertyOf rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#roleOf"/>
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>specRoleOf</rdfs:label>
        <rdfs:comment></rdfs:comment>
 </rdf:Property>

 <rdf:Property rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#genRoleOf">
        <rdfs:subPropertyOf rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#roleOf"/>
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>genRoleOf</rdfs:label>
        <rdfs:comment></rdfs:comment>
```

```
</rdf:Property>

<rdf:Property rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#roleType">
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>roleType</rdfs:label>
        <rdfs:comment></rdfs:comment>
        <rdfs:range rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#RoleClass"/>
</rdf:Property>

<rdf:Property rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#roleSuperClass">
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>roleSuperClass</rdfs:label>
        <rdfs:comment></rdfs:comment>
        <rdfs:domain rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#RoleClass"/>
        <rdfs:range rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#RoleClass"/>
</rdf:Property>

<rdf:Property rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#player">
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>player</rdfs:label>
        <rdfs:comment></rdfs:comment>
</rdf:Property>

<rdf:Property rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#playerClass">
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>playerClass</rdfs:label>
        <rdfs:comment></rdfs:comment>
        <rdfs:range rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#RoleClass"/>
</rdf:Property>

<rdf:Property rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#playerProperty">
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>playerProperty</rdfs:label>
        <rdfs:comment></rdfs:comment>
        <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
</rdf:Property>

<rdf:Property rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#role">
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>role</rdfs:label>
        <rdfs:comment></rdfs:comment>
</rdf:Property>

<rdf:Property rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#roleClass">
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>roleClass</rdfs:label>
        <rdfs:comment></rdfs:comment>
        <rdfs:range rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#RoleClass"/>
</rdf:Property>

<rdf:Property rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#roleProperty">
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>roleProperty</rdfs:label>
        <rdfs:comment></rdfs:comment>
        <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
</rdf:Property>

<rdf:Property rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#propertyRelation">
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>propertyRelation</rdfs:label>
```

```
        <rdfs:comment></rdfs:comment>
        <rdfs:range rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#Relation"/>
  </rdf:Property>

  <rdfs:Class rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#Relation">
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>Relation</rdfs:label>
        <rdfs:comment></rdfs:comment>
  </rdfs:Class>

  <rdfs:Class rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#RoleRelation">
        <rdfs:subClassOf rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#Relation"/>
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>RoleRelation</rdfs:label>
        <rdfs:comment></rdfs:comment>
  </rdfs:Class>

  <rdfs:Class rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#IdentityRelation">
        <rdfs:subClassOf rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#Relation"/>
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>IdentityRelation</rdfs:label>
        <rdfs:comment></rdfs:comment>
  </rdfs:Class>

  <rdfs:Class rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#CounterpartRelation">
        <rdfs:subClassOf rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#Relation"/>
        <rdfs:isDefinedBy rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#"/>
        <rdfs:label>CounterpartRelation</rdfs:label>
        <rdfs:comment></rdfs:comment>
  </rdfs:Class>

</rdf:RDF>
```

A.2 Rollenhierachie

File: http://huemer.lstadler.net/role/rh.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rrdf="http://huemer.lstadler.net/role/rrdf.rdf#">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/rh.rdf#PersonBaseRole">
        <rdf:type rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#RoleClass"/>
</rdf:Description>

<rrdf:RoleClass rdf:about="http://huemer.lstadler.net/role/rh.rdf#EmployeeRole">
        <rrdf:roleSuperClass
rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PersonBaseRole"/>
        <owl:hasKey rdf:parseType="Collection">
                <rdf:Description rdf:about="http://huemer.lstadler.net/role/rrdf.rdf#roleOf"/>
                <rdf:Description rdf:about="http://huemer.lstadler.net/role/rh.rdf#employer"/>
        </owl:hasKey>
</rrdf:RoleClass>

<rrdf:RoleClass rdf:about="http://huemer.lstadler.net/role/rh.rdf#ProjectManagerRole">
        <rrdf:roleSuperClass rdf:resource="http://huemer.lstadler.net/role/rh.rdf#EmployeeRole"/>
</rrdf:RoleClass>
```

```
<rrdf:RoleClass rdf:about="http://huemer.lstadler.net/role/rh.rdf#StudentRole">
        <rrdf:roleSuperClass
rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PersonBaseRole"/>
</rrdf:RoleClass>

<rrdf:RoleClass rdf:about="http://huemer.lstadler.net/role/rh.rdf#PrivatePersonRole">
        <rrdf:roleSuperClass
rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PersonBaseRole"/>
</rrdf:RoleClass>

</rdf:RDF>
```

A.3 Lösung Anwendungsfall 1: Neuerstellung einer Rollenbeschreibung

File: http://huemer.lstadler.net/role/uc1/foaf.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rrdf="http://huemer.lstadler.net/role/rrdf.rdf#"
xmlns:rh ="http://huemer.lstadler.net/role/rh.rdf#"
xmlns:foaf="http://xmlns.com/foaf/0.1/">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc1/foaf.rdf#black">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PersonBaseRole"/>
        <foaf:name> Peter Black </foaf:name>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc1/foaf.rdf#empBlack1">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#EmployeeRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc1/foaf.rdf#black"/>
        <rh:employer rdf:resource="http://www.unixy.net/uni"/>
        <rh:salary>1000</rh:salary>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc1/foaf.rdf#prjBlack">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#ProjectManagerRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc1/foaf.rdf#empBlack1"/>
        <rh:project>SemTech</rh:project>
        <rh:task>Quality Assurance</rh:task>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc1/foaf.rdf#empBlack2">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#EmployeeRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc1/foaf.rdf#black"/>
        <rh:employer rdf:resource="http://www.CompZ.net/"/>
        <rh:salary>1500</rh:salary>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc1/foaf.rdf#stdBlack">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#StudentRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc1/foaf.rdf#black"/>
        <rh:university rdf:resource="http://www.unixy.net/uni"/>
        <rh:subject>Information Sciences</rh:subject>
```

```
        <rh:ID>0356789</rh:ID>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc1/foaf.rdf#privBlack">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PrivatePersonRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc1/foaf.rdf#black"/>
        <rh:task>Washing Up</rh:task>
</rdf:Description>

</rdf:RDF>
```

A.4 Lösung Anwendungsfall 2: Rollen als Spezialisierung

File: http://huemer.lstadler.net/role/uc2/black.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:foaf="http://xmlns.com/foaf/0.1/">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc2/black.rdf#black">
        <foaf:name> Peter Black </foaf:name>
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc2/emp1.rdf"/>
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc2/emp2.rdf"/>
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc2/std.rdf"/>
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc2/priv.rdf"/>
</rdf:Description>

</rdf:RDF>
```

File: http://huemer.lstadler.net/role/uc2/emp1.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rrdf="http://huemer.lstadler.net/role/rrdf.rdf#"
xmlns:rh ="http://huemer.lstadler.net/role/rh.rdf#">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc2/black.rdf#black">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PersonBaseRole"/>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc2/emp1.rdf#empBlack1">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#EmployeeRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc2/black.rdf#black"/>
        <rh:employer rdf:resource="http://www.unixy.net/uni"/>
        <rh:salary>1000</rh:salary>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc2/emp1.rdf#prjBlack">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#ProjectManagerRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc2/emp1.rdf#empBlack1"/>
        <rh:project>SemTech</rh:project>
        <rh:task>Quality Assurance</rh:task>
</rdf:Description>
```

```
</rdf:RDF>
```

File: http://huemer.lstadler.net/role/uc2/emp2.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rrdf="http://huemer.lstadler.net/role/rrdf.rdf#"
xmlns:rh ="http://huemer.lstadler.net/role/rh.rdf#">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc2/black.rdf#black">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PersonBaseRole"/>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc2/emp2.rdf#empBlack2">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#EmployeeRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc2/black.rdf#black"/>
        <rh:employer rdf:resource="http://www.CompZ.net/"/>
        <rh:salary>1500</rh:salary>
</rdf:Description>

</rdf:RDF>
```

File: http://huemer.lstadler.net/role/uc2/std.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rrdf="http://huemer.lstadler.net/role/rrdf.rdf#"
xmlns:rh ="http://huemer.lstadler.net/role/rh.rdf#">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc2/black.rdf#black">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PersonBaseRole"/>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc2/std.rdf#stdBlack">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#StudentRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc2/black.rdf#black"/>
        <rh:university rdf:resource="http://www.unixy.net/uni"/>
        <rh:subject>Information Sciences</rh:subject>
        <rh:ID>0356789</rh:ID>
</rdf:Description>

</rdf:RDF>
```

File: http://huemer.lstadler.net/role/uc2/priv.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rrdf="http://huemer.lstadler.net/role/rrdf.rdf#"
xmlns:rh ="http://huemer.lstadler.net/role/rh.rdf#">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc2/black.rdf#black">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PersonBaseRole"/>
```

```
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc2/priv.rdf#privBlack">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PrivatePersonRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc2/black.rdf#black"/>
        <rh:task>Washing Up</rh:task>
</rdf:Description>

</rdf:RDF>
```

A.5 Lösung Anwendungsfall 3: Rollen als Generalisierung

File: http://huemer.lstadler.net/role/uc3/black.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rrdf="http://huemer.lstadler.net/role/rrdf.rdf#"
xmlns:rh ="http://huemer.lstadler.net/role/rh.rdf#"
xmlns:foaf="http://xmlns.com/foaf/0.1/">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc3/black.rdf#black">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PersonBaseRole"/>
        <foaf:name> Peter Black </foaf:name>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc3/emp1.rdf#empBlack1">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#EmployeeRole"/>
        <rrdf:genRoleOf rdf:resource="http://huemer.lstadler.net/role/uc3/black.rdf#black"/>
</rdf:Description>
<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc3/emp2.rdf#empBlack2">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#EmployeeRole"/>
        <rrdf:genRoleOf rdf:resource="http://huemer.lstadler.net/role/uc3/black.rdf#black"/>
</rdf:Description>
<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc3/emp1.rdf#prjBlack">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#ProjectManagerRole"/>
        <rrdf:specRoleOf rdf:resource="http://huemer.lstadler.net/role/uc3/emp1.rdf#empBlack1"/>
</rdf:Description>
<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc3/std.rdf#stdBlack">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#StudentRole"/>
        <rrdf:genRoleOf rdf:resource="http://huemer.lstadler.net/role/uc3/black.rdf#black"/>
</rdf:Description>
<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc3/priv.rdf#privBlack">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PrivatePersonRole"/>
        <rrdf:genRoleOf rdf:resource="http://huemer.lstadler.net/role/uc3/black.rdf#black"/>
</rdf:Description>

<rdf:Description rdf:parseType="Resource">
        <rrdf:player rdf:resource="http://huemer.lstadler.net/role/uc3/black.rdf#black"/>
        <rrdf:playerProperty rdf:resource="http://huemer.lstadler.net/role/rh.rdf#salary"/>
        <rrdf:role rdf:resource="http://huemer.lstadler.net/role/uc3/emp1.rdf#empBlack1"/>
        <rrdf:role rdf:resource="http://huemer.lstadler.net/role/uc3/emp2.rdf#empBlack2"/>
        <rrdf:roleProperty rdf:resource="http://huemer.lstadler.net/role/rh.rdf#salary"/>
        <rrdf:propertyRelation rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#RoleRelation"/>
</rdf:Description>

<rdf:Description rdf:parseType="Resource">
        <rrdf:playerClass rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PersonBaseRole"/>
        <rrdf:playerProperty rdf:resource="http://huemer.lstadler.net/role/rh.rdf#employer"/>
```

```
        <rrdf:roleClass rdf:resource="http://huemer.lstadler.net/role/rh.rdf#EmployeeRole"/>
        <rrdf:roleProperty rdf:resource="http://huemer.lstadler.net/role/uc3/emp1.rdf#employer"/>
        <rrdf:roleProperty rdf:resource="http://huemer.lstadler.net/role/rh.rdf#employer"/>
        <rrdf:propertyRelation rdf:resource="http://huemer.lstadler.net/role/rrdf.rdf#RoleRelation"/>
</rdf:Description>

</rdf:RDF>
```

File: http://huemer.lstadler.net/role/uc3/emp1.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:un ="http://huemer.lstadler.net/role/uc3/emp1.rdf#"
xmlns:rh ="http://huemer.lstadler.net/role/rh.rdf#">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc3/emp1.rdf#empBlack1">
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc3/black.rdf"/>
        <un:employer rdf:resource="http://www.unixy.net/uni"/>
        <rh:salary>1000</rh:salary>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc3/emp1.rdf#prjBlack">
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc3/black.rdf"/>
        <rh:project>SemTech</rh:project>
        <rh:task>Quality Assurance</rh:task>
</rdf:Description>

</rdf:RDF>
```

File: http://huemer.lstadler.net/role/uc3/emp1.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:rh ="http://huemer.lstadler.net/role/rh.rdf#">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc3/emp2.rdf#empBlack2">
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc3/black.rdf"/>
        <rh:employer rdf:resource="http://www.CompZ.net/"/>
        <rh:salary>1500</rh:salary>
</rdf:Description>

</rdf:RDF>
```

File: http://huemer.lstadler.net/role/uc3/std.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:rh ="http://huemer.lstadler.net/role/rh.rdf#">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc3/std.rdf#stdBlack">
```

```
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc3/black.rdf"/>
        <rh:university rdf:resource="http://www.unixy.net/uni"/>
        <rh:subject>Information Sciences</rh:subject>
        <rh:ID>0356789</rh:ID>
</rdf:Description>

</rdf:RDF>
```

File: http://huemer.lstadler.net/role/uc3/priv.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:rh ="http://huemer.lstadler.net/role/rh.rdf#">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc3/priv.rdf#privBlack">
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc3/black.rdf"/>
        <rh:task>Washing Up</rh:task>
</rdf:Description>

</rdf:RDF>
```

A.6 Lösung Anwendungsfall 4: Anreicherung von vorhandenen Ressourcen

File: http://huemer.lstadler.net/role/uc4/black.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:foaf="http://xmlns.com/foaf/0.1/">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc4/black.rdf#black">
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc4/myroles.rdf"/>
        <foaf:name> Peter Black </foaf:name>
</rdf:Description>

</rdf:RDF>
```

File: http://huemer.lstadler.net/role/uc4/emp1.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:rh ="http://huemer.lstadler.net/role/rh.rdf#">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc4/emp1.rdf#empBlack1">
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc4/myroles.rdf"/>
        <rh:employer rdf:resource="http://www.unixy.net/uni"/>
        <rh:salary>1000</rh:salary>
</rdf:Description>
```

```
<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc4/emp1.rdf#prjBlack">
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc4/myroles.rdf"/>
        <rh:project>SemTech</rh:project>
        <rh:task>Quality Assurance</rh:task>
</rdf:Description>

</rdf:RDF>
```

File: http://huemer.lstadler.net/role/uc4/emp2.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:rh ="http://huemer.lstadler.net/role/rh.rdf#">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc4/emp2.rdf#empBlack2">
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc4/myroles.rdf"/>
        <rh:employer rdf:resource="http://www.CompZ.net/"/>
        <rh:salary>1500</rh:salary>
</rdf:Description>

</rdf:RDF>
```

File: http://huemer.lstadler.net/role/uc4/std.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:rh ="http://huemer.lstadler.net/role/rh.rdf#">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc4/std.rdf#stdBlack">
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc4/myroles.rdf"/>
        <rh:university rdf:resource="http://www.unixy.net/uni"/>
        <rh:subject>Information Sciences</rh:subject>
        <rh:ID>0356789</rh:ID>
</rdf:Description>

</rdf:RDF>
```

File: http://huemer.lstadler.net/role/uc4/priv.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:rh ="http://huemer.lstadler.net/role/rh.rdf#">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc4/priv.rdf#privBlack">
        <rdfs:seeAlso rdf:resource="http://huemer.lstadler.net/role/uc4/myroles.rdf"/>
        <rh:task>Washing Up</rh:task>
</rdf:Description>

</rdf:RDF>
```

File: http://huemer.lstadler.net/role/uc4/myroles.rdf

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:rrdf="http://huemer.lstadler.net/role/rrdf.rdf#">

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc34/black.rdf#black">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PersonBaseRole"/>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc4/emp1.rdf#empBlack1">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#EmployeeRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc4/black.rdf#black"/>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc4/emp2.rdf#empBlack2">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#EmployeeRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc4/black.rdf#black"/>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc4/emp1.rdf#prjBlack">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#ProjectManagerRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc4/emp1.rdf#empBlack1"/>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc4/std.rdf#stdBlack">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#StudentRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc4/black.rdf#black"/>
</rdf:Description>

<rdf:Description rdf:about="http://huemer.lstadler.net/role/uc4/priv.rdf#privBlack">
        <rrdf:roleType rdf:resource="http://huemer.lstadler.net/role/rh.rdf#PrivatePersonRole"/>
        <rrdf:roleOf rdf:resource="http://huemer.lstadler.net/role/uc4/black.rdf#black"/>
</rdf:Description>

</rdf:RDF>
```

A.7 Dokumentation der Änderungen im Sourcecode von Tabulator

Im Folgenden sollen die vorgenommenen Änderungen und Ergänzungen im Quellcode von Tabulator, wodurch Rollen ihrer Semantik entsprechend dargestellt werden können, dokumentiert werden. Für eine genauere Beschreibung der Funktionen sei an dieser Stelle auf den Quellcode verwiesen.

Die Änderungen und Ergänzungen beschränken sich auf vier Dateien, nämlich auf tabulate.js, outline.js, until.js und die defaultPane.js.

tabulate.js

Die tabulate.js ist das Top-Level-Skript-File welches unter anderem Definitionen von globalen Variablen beinhaltet. Hier sind folgende Änderungen durchgeführt worden:

- Hinzufügen von globalen Variablen für Namensräume (rrdf)
- Hinzufügen von globalen Variablen für Rollenicons
- Hinzufügen von globalen Variablen für Rollenicontitel
- Hinzufügen der globalen Variable roleLevel

defaultPane.js

Diese Datei beinhaltet Funktionen und Funktionsaufrufe zum Darstellen von RDF-Graphen in der Standardansicht. Hier sind folgende Änderungen durchgeführt worden:

- Adaptieren der Prädikatliste in der Funktion render, sodass aufwärts vererbte Prädikate in die Darstellung miteinbezogen werden

outline.js

Diese Datei beinhaltet die eigentlichen Darstellungsfunktionen. Hier sind folgende Änderungen durchgeführt worden:

- Implementieren der Funktion appendRoleIcon
- Adaptieren der Funktion outline_predicateTD
- Adaptieren der Funktion propertyTR
- Implementieren der Funktion rolePropertyTR
- Implementieren der Funktion appendRoleTRs
- Adaptieren der Funktion appendPropertyTRs
- Adaptieren der Funktion TabulatorMousedown
- Adaptieren der Funktion outline_expand
- Implementieren der Funktion outline_role_expand
- Implementieren der Funktion outline_role_collapse

util.js

Diese Datei beinhaltet Vorbereitungsfunktionen für die Darstellung. Hier sind folgende Änderungen durchgeführt worden:

- Implementieren der Funktion labelForRoleProperty
- Adaptieren der Funktion predicateLabelForXML
- Implementieren der Funktion isKeyProperty

- Implementieren der Funktion getRoleClass
- Implementieren der Funktion getKeys
- Hinzufügen der Funktion generateInheritedProperties

  Diese Funktion ist nicht implementiert und dient als Ansatzpunkt um Generalisierungsinformationen (vgl. Kapitel 5.1.2 und 5.1.3) im Tabulator zu interpretieren. Die Einbindung dieser Funktion in den Darstellungsablauf hat bereits erfolgt. Die Funktion muss nur noch implementiert werden und die für die Darstellung relevanten Prädikate als return-Wert liefern. Für die Beschreibung der Ein- und Ausgangsparameter sei auf die Kommentare im Sourcecode verwiesen.
- Implementieren der Funktion mergeLists
- Implementieren der Funktion cutPropertyList
- Implementieren der Funktion isRole
- Implementieren der Funktion isRoleProperty
- Implementieren der Funktion getRoleList
- Implementieren der Funktion getRolePropertyList
- Implementieren der Funktion sortPredicateList

A.8 Für Entwickler: Eine kurze Einführung in den Sourcecode von Tabulator

Um Entwicklern, die Tabulator erweitern möchten, einen schnelleren Einstieg zu geben sollen ein paar wichtige Funktionen als Einstiegspunkte erklärt werden. Eins sei vorweg schon gesagt, nämlich dass der Sourcecode von Tabulator zum Teil sehr schwer lesbar ist und dadurch eine sehr lange Einarbeitungszeit entsteht. Es sind die Möglichkeiten und Freiheiten in Javascript zum Leide der Lesbarkeit sehr weit ausgeschöpft worden.

Die zentrale HTML-Datei ist tab.html. In diese Datei werden alle Javascript Dateien inkludiert. Weitere wichtige Dateien für den Erkundungsmodus (vgl. Kapitel 6.3.1), sind outline.js, util.js, term.js, identity.js und source.js. In der outline.js befinden sich Funktionen zur Steuerung der View. Die util.js bietet Funktionen zur Unterstützung der Darstellungsfunktionen in der outline.js. Die term.js beinhaltet Klassendefinitionen welche die RDF-Struktur abbilden (beispielsweise die Klassen RDFSubject, RDFProperty, RDFStatement, …). Die identity.js beinhaltet die Klassendefinition des internen Datastore.

Die source.js beinhaltet unter anderem Funktionen um RDF-Dokumente anzufragen und zu verarbeiten.

Der erste wichtige Funktionsaufruf nach der Bestätigung der Eingabe einer URI vom Benutzer findet in der outline.js statt. Es wird dort die Funktion GotoFormURI_enterKey und gleich darauf die Funktion GotoFormURI aufgerufen. Weitere wichtige Funktionen im Erkundungsmodus sind:

- outline_expand: Dies ist die zentrale Funktion zum Zeichnen eines Baumes. Dieser wird immer dann aufgerufen, wenn zu einem Subjekt oder Objekt die Prädikate und Objekte dargestellt werden sollen. Dies ist dann der Fall, wenn ein Benutzer eine URI anfragt und die Dereferenzierung dieser Informationen liefert, oder wenn auf das expand-Icon (graues kleines dreieckiges Icon) einer Subjekt-URI oder Objekt-URI im Baum geklickt wird.
- outline_collapse: Diese Funktion ist das Gegenstück zur Funktion outline_expand und klappt einen Baum zusammen. Diese Funktion wird aufgerufen, wenn auf das kleine graue collapse-Icon neben einem Subjekt oder Objekt geklickt wird.
- outline_objectTD: Diese Funktion erstellt die Darstellung eines Objektes eines RDF-Statements in HTML.
- outline_predicateTD: Diese Funktion erstellt die Darstellung eines Prädikates eines RDF-Statements in HTML.
- appendPropertyTRs: Diese Funktion wird unter anderem von outline_expand aufgerufen und kümmert sich um die Darstellung eines Prädikat-Objekt-Paares zu einem Subjekt. Diese Funktion ruft unter anderem outline_predicateTD und outline_objectTD auf.
- appendRoleTRs: Diese Funktion wird von appendPropertyTRs aufgerufen, wenn ein Rollenprädikat vorkommt, d.h. wenn eine Rolle mit möglicherweise Subrollen dargestellt werden soll.
- addButtonCallbacks: Diese Funktion registriert Funktionen die bei Ereignissen (beispielsweise ‚done' oder ‚fail' eines URI-Requests) gefeuert werden sollen. Die registrierten Funktionen sind Aktualisierungsfunktionen für bestimmte Objekte der View.
- outline_role_expand: Diese Funktion wird aufgerufen, wenn eine Rolle ausgeklappt werden soll, also wenn auf das kleine dreieckige Icon neben der Rolle geklickt wird.

- outline_role_collapse: Diese Funktion wird aufgerufen, wenn eine Rolle zusammengeklappt werden soll, also wenn auf das kleine dreieckige Icon neben einer Rolle geklickt wird.

Eine wichtige Funktion in der source.js ist:

- lookUpThing: Diese Funktion wird aufgerufen, wenn eine URI requested werden soll. Sie initiiert den Dereferenziervorgang einer URI.

Wichtige Funktionen in der until.js sind:

- isKeyProperty: Diese Funktion überprüft ob ein Prädikat ein Schlüssel einer qualifizierten Rolle ist.
- isRole: Diese Funktion überprüft ob eine URI als Subjekt in einem Rollenstatement vorkommt.

Eine wichtige und hilfreiche Funktion befindet sich in der identity.js:

- statementsMatching: Mit dieser Funktion können Statements aus dem internen Datastore (vgl. Kapitel 6.2.2) gefiltert werden.