aLETHEIa. Semantische Föderation umfassender Produktinformationen

D.G4.5
Datenhistorien und Rückverfolgbarkeit

Verantwortlicher Partner	Sandro Reichert, TU Dresden
Vorhabensbezeichnung	Aletheia
Förderkennzeichen	01IA08001
Version	0.9
Datum	27.01.2010

GEFÖRDERT VOM

Bundesministerium für Bildung und Forschung

Dokumenteninformationen	
Typ	Deliverable
Kennzeichner	D.G4.5 Datenhistorien und Rückverfolgbarkeit
Fälligkeitstermin	31.01.2010
Arbeitspaket	AP.G4 Verwaltung und Föderation von Produktinformationen
Verantwortlicher Partner	Sandro Reichert, TU Dresden

Reviewer	
Peer Reviewer 1	Magnus Niemann, FU Berlin
Peer Reviewer 2[1]	

Kontaktinformationen der Verfasser			
Name	**Organisation**	**Email**	**Tel**
Sandro Reichert	TUD	sandro.reichert@tu-dresden.de	+49-351-46338235
Anne Becker	Ontoprise GmbH	anne.becker@ontoprise.de	+49-721-50980933
Robert Rieger	SAP AG	robert.rieger@sap.com	+49-351-48116212

Zugehörige Dokumente			
Datum	**Name des Dokuments**	**Verfasser**	**Kommentare**

[1] Falls abweichend von Reviewer 1

Versionierung und Beitragshistorie[2]					
Version	**Status**	**Verfasser[3] (Name, Organisation)**	**Fälligkeit [4]**	**Ist-Datum**	**Kommentar**
0.01	Neu	Sandro Reichert		05.05.2009	
0.10	Initialer Draft	Sandro Reichert	27.10.2009	07.10.2009	>=40 Arbeitstage vor Abgabetermin
0.2	Konsolidierte Version	Deliverable-Lead	06.01.2010	15.01.2010	
0.30	Freigabe für Review 1	Deliverable-Lead	11.01.2009	18.01.2010	1. des Abgabemonats
0.35	Review 1: Kommentierte Version	Peer Reviewer 1	18.01.2010	23.01.2010	8. des Abgabemonats
0.4	Ergänzungen der Anwendungspartner	Sandro Reichert		25.01.2010	
0.8	Final konsolidierte und editierte Version	Deliverable-Lead	27.01.2010	25.01.2010	27. des Abgabemonats
0.9	Freigegebene Version	Deliverable-Lead	29.01.2010	27.01.2010	
1.0	An DLR gesendet	PMO	29.01.2010		29. des Abgabemonats

[2] Siehe: http://87.106.216.160/aletheia/index.php/Deliverables_Erstellung_und_Review und http://87.106.216.160/aletheia/index.php/Deliverables_Namenskonventionen

[3] Bitte durch Person und Organisation ersetzen

[4] Relative Angaben bitte durch deliverable-spezifische Daten ersetzen

Danksagung / Hinweis

Die hier beschriebenen Arbeiten wurden teilweise mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01IA08001 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor/bei den Autoren.

Inhaltsverzeichnis

1 Einführung

1.1 Motivation

Das Aletheia-Projekt hat es sich zum Ziel gesetzt, produktbezogene Informationen aus heterogenen Quellen zu integrieren und für einen breiten Anwenderkreis verfügbar zu machen. Ein Aspekt ist, es dem Nutzer des Systems zu ermöglichen, die Relevanz der dargestellten Informationen zu bewerten. Im Projektantrag wird vorgeschlagen, zu den Produktinformationen zusätzliche Metadaten zu Qualität, Frische und Herkunft zu erfassen, zu verwalten und dem Nutzer bereitzustellen.

Seit ca. 4000 Jahren organisieren Menschen Informationen, um sie später wiederzufinden und wieder zu verwenden. Typische Beispiele sind die bereits seit Hunderten von Jahren existierenden Inhaltsverzeichnisse in Büchern oder Indizes. Der Einsatz von Metadaten zur Kategorisierung von Datenbeständen und zur Suche in diesen stammt ursprünglich aus dem Bibliothekswesen, wo bereits seit Jahrhunderten eine Suche über Metadaten wie Autoren möglich ist [BYRN99]. Ein weiteres typisches Beispiel ist in wissenschaftlichen Veröffentlichungen oder Forschungsberichten zu finden. Informationen, die nicht vom Autor selbst stammen, werden mit der Angabe von Quellen untermauert. Basierend auf der Angabe der Autoren, Titel der Quelle, Datum der Erscheinung sowie die Zuordnung zu einer Konferenz oder einem Buch kann die Quelle zumeist eindeutig identifiziert werden, was dem Leser die Möglichkeit bietet, die wiedergegeben Informationen zu ihrem Ursprung zurück zu verfolgen und ihre Qualität und Aktualität zu Bewerten.

Das im Aletheia-Projekt entstehende System führt Informationen aus heterogenen Quellen zusammen und präsentiert sie dem Anwender in einer einheitlichen Weise. Im Unterschied zu einer Bibliothek oder einem klassischen Index, in den „nur“ auf vorhandene Daten verwiesen wird, werden die Informationen extrahiert, wobei je nach Struktur der Quelle ein Informationsverlust und Fehlinterpretationen auftreten können. Diese Informationen müssen in Form von Qualitätsmetadaten festgehalten, verwaltet und bei Suchanfragen in die Berechnung des Ergebnisses einbezogen werden.

1.2 Aufbau des Deliverables

Der Aufbau des Deliverables folgt der Aufteilung der Subtasks des Projektantrages. In Kapitel 2 erfolgt zunächst eine Identifikation der zu betrachtenden Qualitätsmetadaten und ihrer Funktionalitäten. Der Schwerpunkt ist eine ausführliche Analyse der Typen von Metadaten, die in Aletheia verwendet werden. Kapitel 3 beschreibt allgemeine und konkrete Methoden zur Gewinnung von Qualitätsmetadaten und beleuchtet die Erweiterungen der Datenintegrationsplattform. Um die gewonnen Metadaten in Aletheia zu verwalten folgen in Kapitel 4 Grundlagen und eine State-of-the-Art-Betrachtung zur Verwaltung von Metadaten. Das wichtigste Unterscheidungsmerkmal der Metadaten ist, ob sie sicher oder unsicher sind. Eine Betrachtung der Methoden zur Nutzung von Qualitätsmetadaten in Kapitel 4 rundet das vorliegende Dokument ab. Als Nutzer der Metadaten werden sowohl die Nutzer des Aletheia-Systems sowie das System selbst betrachtet.

2 Identifikation der Qualitätsmetadaten und Funktionalitäten

In diesem Kapitel werden zuerst die Anforderungen und Wünsche aus Sicht der Anwendungspartner aufgeführt (2.1), gefolgt von einigen Begriffsdefinitionen in Abschnitt 2.2. Anschließend werden die Wünsche der Anwender konkretisiert (2.3). In Abschnitt 2.4 werden die in T.G4.5 zu betrachtenden Qualitätsmetadaten ermittelt und ausführlich vorgestellt.

2.1 Anforderungen der Anwender

Die gebräuchlichste Verwendung von Metadaten in aktuellen Systemen besteht in der Filterung oder dem Ranking von Ergebnissen anhand von Metadaten wie einem (Kalender-)Datum. Diese Funktionalität wird daher auch vom Aletheia-System erwartet.

Im Internet der Dinge werden Daten zu Produkten hauptsächlich auf Item-Ebene erfasst. Das bedeutet, dass das einzelne, physisch existente Produkt im Mittelpunkt der Aufmerksamkeit steht. Diese Produkte bewegen sich (bspw. im Anwendungsszenario AP.A3) in Produktverpackungen und/oder Containern allein oder in Gruppen durch logistische Prozesse. Darüber hinaus können neben den Produkten weitere Objektklassen existieren, die als Elemente des Internets der Dinge Lieferanten von Informationen sein können. Als Beispiel können Personen genannt werden, die mit den Produkten (Objekten) interagieren. Im Internet der Dinge kommt es aufgrund der angesprochenen Orientierung auf materielle Objekte dazu, dass die hauptsächlich relevanten Informationen zu den betrachteten Objekten sich hauptsächlich auf die Dimensionen Ort, Zeit und Zustand beziehen. Sie können daher als Metadaten verstanden werden. Im Idealfall soll zu jeder Zeit bekannt sein, wo sich das Objekt befindet und wie es dem Objekt geht.

Zu diesen Dimensionen (wobei die Dimension Zustand ihrerseits bereits komplex sein kann), sollen in der AP.G2-Architektur auch Informationen darüber erfasst werden, welche Güte (Präzision) die Messung des Orts/Zeitpunkts/Zustands besaß. Die Güte der erfassten Metadaten entspricht demnach Metametadaten. Eine Erfassung der Vertrauenswürdigkeit einer Messung wird ebenfalls vorgesehen, da das im Arbeitspaket AP.A3 entstehende System zur Überwachung der Qualität von Transportstrecken konzipiert wird, was die Frage nach der Manipulierbarkeit und somit Sicherheit und Vertrauenswürdigkeit impliziert.

Über die grundsätzliche Erfassung der Zustände hinaus ist die Intention hinter dem in A3 entstehenden System die eines Überwachungssystems. Zu den verschiedenen Dimensionen (Ort, Zeit, Zustand) gibt es einen entsprechenden SOLL-Prozess. Das System hat das Ziel, die Einhaltung des SOLL-Prozesses in Transportsystemen für logistische Güter zu gewährleisten. Das System detektiert Abweichungen des IST- vom SOLL-Prozess und generiert darauf eine Information (Alarm-Event), die an anderer Stelle des System Gegenmaßnahmen auslöst. Diese Informationen sind ebenfalls der Anforderung der Vertrauenswürdigkeit unterworfen und werden unmittelbar durch fehlende Präzision in der Messung der Kenngrößen negativ beeinflusst (Es kommt ggf. zu Fehlalarmen oder ausbleibenden Alarmmeldungen).

Als zusätzliche Eigenschaft des Systems ist es möglich, je nach Anforderung an das System in Echtzeit, die genannten Zustandsdimensionen auch über den Verlauf des Transportwegs zu protokollieren, um im Nachgang bspw. statistische Auswertung über die Datenhistorien legen zu können. Dies sei an einem Beispiel illustriert: Ein Produkt (physisches Objekt, in

einer Verpackung, Karton) bewegt sich im logistischen Prozess. Die letzte Erfassung des Aufenthaltsortes liegt 2 Tage zurück (Frische). Dabei wurde mit klassischem RFID eine Lokalisierungstechnologie verwendet, die nicht besonders genau war, das gesuchte Produkt befindet sich im Umkreis von 12 Metern (Genauigkeit) um den ermittelten Punkt. Eine Rückverfolgbarkeit des Produktes ist über die Historie der Positionsdaten gewährleistet, über die der Transportweg nachvollzogen werden kann.

In Anforderung AP.A2-AF97 wird eine Versionierung der Repositorien gefordert. Die State-of-the-Art-Analyse in D.G4.2 hat jedoch gezeigt, dass keines der untersuchten Tools die Versionierung unterstützt. Eine aufwendige Erweiterung bestehender Technologien um Techniken zur Versionierung ist nicht Bestandteil des Aletheia-Projektes, sodass keine Metainformationen zur Versionierung vorgehalten werden müssen.

Granularität der erfassten Daten

Exemplarisch lässt sich der Begriff der Granularität im Internet der Dinge erläutern: Aus Kostengründen kann es bspw. notwendig sein, lediglich einen ganzen ISO-Container (20 oder 40 Fuß lang) mit einem Gerät zur Temperaturüberwachung auszustatten. Diese Temperaturmessungen repräsentieren dann die Temperatur, wie sie für den gesamten Container angenommen wird. Die Granularität der Datenerfassung ist somit nicht hoch. Alternativ können alle Produktverpackungen im Container mit einem Sensorknoten ausgestattet sein und jeder von ihnen kann die lokale Temperatur messen. Die Granularität der Information über die Temperatur im gleichen Raumbereich ist deutlich höher: Die Realität wird präziser wiedergegeben.

In den Arbeitspakten AP.A1 und AP.A4 wird ein Mix aus grober und feiner Granularität der Metadaten als sinnvoll erachtet. So sollen auf der einen Seite Dokumente mit Metatags versehen werden können, auf der andern Seite sollen Informationsquellen wie Datenbanken als Ganzes mit Metatags annotiert werden können. Bei der Nutzung der Metadaten sind die für die Dokumente und Informationsquellen bestehenden Zugriffsrechte zu beachten, sodass einige Metadaten nur beschränkt zugreifbar sind, andere wiederum allen Nutzern des Systems zur Verfügung stehen.

Dateien werden typischerweise in projektorientierten Verzeichnisstrukturen abgelegt, sodass ein Dokument X, dass in einem Ordner "Projekt Y" liegt, auch höchstwahrscheinlich von diesem Projekt Y handelt. Diese Information liegt unter Umständen nicht noch einmal explizit in der Datei X selbst vor, sodass man nur über den Pfad eine Zuordnung zum Projekt herstellen kann. Diese Information darf bei der Extraktion nicht verloren gehen. Wie auch in allen anderen Anwenderarbeitspaketen müssen auch hier die Zugriffsrechte der Daten auf die extrahierten Informationen im Repository abgebildet werden.

Ein weiterer Wunsch eines Anwendungspartners ist die Suche nach dem ersten Vorkommen einer bestimmten Information, um ihren Urheber ausfindig zu machen. So ist es betrieblichen Alltag üblich, Inhalte per Copy&Paste zwischen verschiedenen Präsentationsmaterialien zu kopieren. Enthält eines der Dokumente oder gar die Quelle selbst einen Fehler, so muss der Urheber gefunden werden können. Es wäre eine Erleichterung, wenn Aletheia den Nutzer bei der Suche unterstützte, indem es Dokumente

fände, welche gleiche oder ähnliche Informationen enthielten und zu diesen Dokumenten das Datum der letzten Änderung angäbe.

In den Anforderungen der Anwender wird deutlich, dass sich die gewünschten Metadaten sowohl auf die Produkte selbst als auch auf die Qualität der Produktdaten beziehen. Aus diesem Grund folgt im nächten Abschnitt eine ausführliche der zu verwendenden Begriffe.

2.2 Begriffsklärung

2.2.1 Datenquelle und Ressource

Der in D.G3.1 [SWR09] bereits eingeführte Begriff **Datenquelle** wird im vorliegenden Deliverable auf Quellen wie File-Server oder Datenbanken beschränkt, die von den Data und Information Providern angesprochen werden und die unterste Ebene der in [WSM+09] vorgestellten Architektur bilden (vgl. auch Abbildung 7, S. 34). Der Begriff des abstrakten Dokuments wird durch **Ressource**[5] ersetzt, da z.B. eine Datenbank oder ein Verzeichnis eines Dateisystems im Allgemeinen nicht als Dokument, sondern als Ressource bezeichnet werden.

2.2.2 Daten, Informationen und Metadaten

Der weitegefasste Sammelbegriff Daten wurde in Aletheia bereits im Deliverable D.G4.2 [BWR+09] konkretisiert und wird an dieser Stelle weiter verfeinert.

Primärdaten sind Daten, die in unverarbeiteter Form in den Datenquellen vorliegen und dem Aletheia-System von den Data und Information Providern zur Verfügung gestellt werden. Sofern die enthaltenen **Informationen** nicht bereits von den Providern extrahiert werden, folgt dieser Schritt im Data Integration Layer.

Ein wichtiges Unterscheidungsmerkmal von extrahierten Informationen ist, ob diese korrekt sind, oder nicht. Korrektheit bezieht sich in dem in Aletheia betrachteten Fall ausschließlich auf den Extraktionsprozess, d.h. entspricht die extrahierte, explizit gemachte Information zu 100% der Aussage, die implizit in den bearbeiteten Primärdaten steht. Wenn dies der Fall ist, wird die extrahierte Information als Fakt bezeichnet, andernfalls als unsichere Information (Uncertain Information). Unsichere Informationen wurden in [BWR+09] mit Fuzzy Facts übersetzt, in [WSM+09] wurde die Bezeichnung in Uncertain Information geändert. Als Fuzzy Facts werden Aussagen wie „es ist warm“ verstanden. Hier liegt eine Unschärfe im Ausdruck vor, da warm z.B. einen Temperaturbereich von 20-25°C bezeichnet. Man ist sich also zu 100% sicher, dass es warm ist, kennt jedoch nicht die genaue Temperatur. Bei der in Aletheia durchgeführten Informationsextraktion aus semi- und unstrukturierten Quellen ist hingegen die Unsicherheit des Extraktionsprozesses von Bedeutung, die angibt, wie sicher sich das System ist, dass die extrahierte Information der in den Primärdaten enthaltenen Aussage entspricht. Die Begriffe Aussage und Information werden in Aletheia synonym verwendet.

Metadaten sind maschinenlesbare Daten über andere Daten. Sie können sowohl in den Primärdaten enthalten sein, als auch vom Aletheia-System und dessen Nutzern im Laufe der Verarbeitung gewonnen werden. Eine Unterscheidung zwischen Daten und Metadaten ist

[5] Mit Ressource ist hier der ursprüngliche Begriff einer Ressource gemeint und nicht die im Kontext des Semantic Web verwendete Abkürzung einer RDF-Ressource.

nicht ohne weiteres möglich. Metadaten sind immer nur in einer Beziehung zu anderen Daten als Metadaten zu identifizieren, wie das folgende Beispiel verdeutlicht: Betrachtet man ein Produkt und seine Rezension bei Amazon, dann sind beides für sich Daten, die Rezension enthält Informationen über das Produkt und ist in dieser Beziehung ein Metadatum. Betrachtet man nun diese Rezension und ihre Bewertung durch andere Nutzer, dann ist die Rezension das Datum und die Bewertung das Metadatum.

Bei Metadaten über Metadaten spricht man auch von Metametadaten. Metametadaten werden primär verwendet, um die Qualität oder die Herkunft eines Satzes von Metadaten zu beschreiben. Die Menge der (Klassen von) Metametadaten ist per se beschränkt, da Metametadaten auf einem auf sich selbst referenzierenden Modell basieren: Es sind Informationen über Informationen. Im vorliegenden Dokument bezeichnet der Begriff Metadaten[6] daher sowohl alle Informationen, die sich auf Primärdaten oder aus ihnen extrahierte Produktinformationen beziehen, als auch Informationen über diese Informationen. Es wird nicht zwischen Metadaten und Metametadaten unterschieden.

Abbildung 1 illustriert, dass die Menge aller Informationen die Vereinigung der disjunkten Mengen unsicherer Informationen und (sicherer) Fakten ist. Fakten werden in Aletheia im Semantic Repository vorgehalten, unsichere Informationen im Uncertain Repository [BWR+09]. Metadaten sind eine echte Teilmenge der Informationen, ihre Aufspaltung in unsichere und sichere Metadaten wird in Abschnitt 2.4 näher betrachtet.

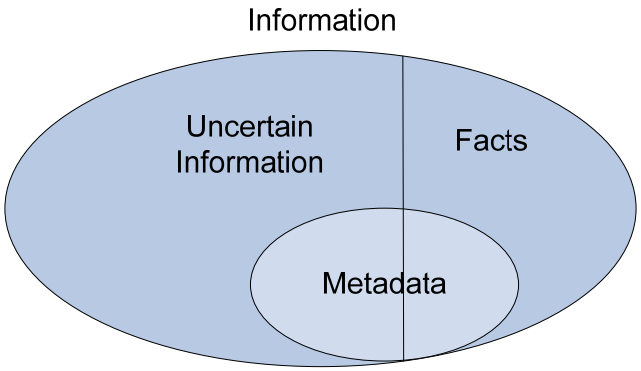

Abbildung 1: Klassifikation von Informationen

2.2.3 Qualität, Datenqualität und Informationsqualität

Die Qualitätsmanagementnorm ISO 9000:2005 definiert **Qualität** als „Grad, in dem ein Satz inhärenter Merkmale Anforderungen erfüllt“. Dabei bedeutet „inhärent“ „der Sache oder dem entsprechenden Merkmal innewohnend“. Metadaten, die eine qualitative Aussage über Merkmale machen, müssen ergo ebenfalls über bestimmte Anforderungen messbar sein. Die im Antrag verwendeten Begriffe: Frische, Historie und Herkunft sind also ohne weitere

[6] Aufgrund unserer Unterscheidung von Daten und Informationen müsste man konsequenterweise die im Repository vorgehaltenen Metadaten als Metainformationen bezeichnen, da sie Informationen über andere Informationen darstellen. Der Begriff Metadaten ist jedoch sehr verbreitet, sodass wir ihn in beiden Fällen verwenden.

Verfeinerung keine Qualitätsmetadaten i.S.v. ISO9000:2005, da sie keine Anforderungen definieren, welche es qualitativ zu erfüllen gäbe. Qualität kann über die Formel in Abbildung 2 berechnet werden, wobei M ein Satz von objektiv messbaren Merkmalen für die Informationseinheit I ist.

$$Q_I = \sum_{m \in M}^{|M|} \frac{IST(m(I))}{|M| \cdot SOLL(m(I))}$$

Abbildung 2: Formel zur Berechnung der Qualität

Jedes Merkmal $m \in M$ definiert objektiv messbare Anforderungen (SOLL) und muss zur Laufzeit quantifizierbar, messbar sein (IST).

Das Thema **Datenqualität** wird vor allem im Bereich der Datenbanken seit Jahrzehnten untersucht. So ergibt sich nach Wang und Strong [WS96] die folgende Kategorisierung von Datenqualität:

Kategorie	Dimension
Intrinsische Datenqualität	Genauigkeit, Glaubhaftigkeit, Objektivität, Reputation
Kontextuelle Datenqualität	Vollständigkeit, Relevanz, Aktualität, Datenmenge, Mehrwert
Repräsentative Datenqualität	Interpretierbarkeit, Verständlichkeit, Konsistenz, Prägnanz
Zugriffsqualität	Zugänglichkeit, Sicherheit

Tabelle 1 : Klassifikation Datenqualität nach [WS96]

Die in Tabelle 2 enthaltene alternative Klassifikation unterscheidet nach inhaltsbezogenen, technischen, subjektiven und instanzbezogenen Merkmalen der Datenqualität [Nau02].

Kategorie	Dimension
Inhaltsbezogene Merkmale	Genauigkeit, Vollständigkeit, Relevanz, Interpretierbarkeit
Technische Merkmale	Verfügbarkeit, Antwortzeit, Latenz
Subjektive Merkmale	Glaubwürdigkeit, Reputation
Instanzbezogene Merkmale	Datenmenge, Verständlichkeit, Verifizierbarkeit

Tabelle 2: Klassifikation nach [Nau02]

Bei genauerer Betrachtung der in den Klassifikationen betrachteten Dimensionen wird deutlich, dass sie sich – wie der Name Datenqualität sagt – auf die Qualität der in einem System enthaltenen Daten beziehen. Aus dem Blickwinkel von Aletheia bedeutet dies, dass sich diese Metadaten auf die in den Datenquellen enthaltenen Primärdaten beziehen. Ziel von Aletheia ist es, die in den Primärdaten enthaltenen Produktinformationen zu extrahieren und semantisch zu föderieren. Hierzu gehört nicht, die Qualität der Primärdaten zu ermitteln oder gar zu verbessern. Metadaten zur Datenqualität können z.B. die Genauigkeit eines Temperatursensors sein, wobei Genauigkeit wieder im Sinne von Genauigkeit der Messung, z.B. Abweichung/Toleranz in °C gesehen werden kann. Qualitätsdaten dieser Art werden in Aletheia wie Produktinformationen selbst betrachtet.

2.2.4 Qualitätsmetadaten

Fokus dieses Deliverables sind Metadaten zur Beschreibung der Qualität der aus den Primärdaten extrahierten Informationen. Die im Projektantrag als **Qualitätsmetadaten** beschriebenen Informationen zur Qualität, Frische und Herkunft beziehen sich im Task T.G4.5 ausschließlich auf die Produktdaten bzw. die aus ihnen extrahierten Produktinformationen. Es werden keine Dinge wie der Geschmack, die Haltbarkeit oder die Lieferkette von Nahrungsmittel betrachtet. Metadaten zur Qualität liefern die in Abschnitt 2.2.3 beschriebenen Eigenschaften, Metadaten zur „Frische“ geben Aufschluss über die Aktualität der im Repository vorgehaltenen Produktinformation und Metadaten zur Herkunft sind Informationen über die Quelle oder den Urheber der Information. Wie bereits ersichtlich ist, sind die Qualitätsmetadaten Qualität, Frische, Herkunft nur Oberklassen von Metadaten. Sie werden in Abschnitt 2.4 verfeinert.

2.3 Überarbeitete Anforderungen der Anwendungspartner

Die in Abschnitt 2.1 gesammelten Anforderungen der Anwendungspartner werden jetzt anhand der soeben erläuterten Begriffe überarbeitet und ergänzt. Aufgrund der zahlreichen Überschneidungen der Anforderungen erfolgt in diesem Abschnitt keine Zuordnung zwischen Anforderungen und Anwendungspartnern.

Die im Repository vorgehaltenen Produktinformationen können aus der subjektiven Sicht der Anwender wahr, falsch oder unüberprüfbar sein, d.h. die Bewertung kann von Person zu Person verschieden ausfallen. Um ihn bei dieser Entscheidung und der Bewertung der Relevanz verschiedener Informationen so gut wie möglich zu unterstützen, wird die Angabe der Informationsquelle gewünscht. Informationsquelle hat hier verschiedene Bedeutungen. Zum einen ist es die Ressource, aus der Aletheia die Information extrahiert hat, sowie die Datenquelle, in der die Ressource vorgehalten wird. Zum anderen ist es der Autor, welcher die Information (digital) abgespeichert hat. Unter Autor sind sowohl Menschen zu verstehen, als auch technische Geräte wie z.B. Sensoren in einem Wireless Sensor Network. Einer Informationsquelle kann je nach Kontext und Informationsempfänger ge- oder misstraut werden. Über die Informationsquelle kann überdies ggf. bekannt sein, dass die Schärfe der Aussage (technisch) bedingt limitiert ist. Jede Information wurde zu einem bestimmten Zeitpunkt generiert und besitzt somit ein definiertes Alter. Dem Zeitpunkt der Generierung kann je nach Kontext, Informationsquelle und Informationsempfänger ge- oder misstraut werden.

Da Aletheia aus verschiedenen Quellen stammende Informationen föderiert, müssen Duplikate erkannt werden und unter anderem die verschiedenen Vertrauenswürdigkeiten der Quellen sowie die Präzessionen des Extraktionsprozesses aggregiert werden.

2.4 Metadaten in Aletheia

2.4.1 Vorbetrachtung

Aus den gesammelten Anforderungen der Anwender werden nun die in Aletheia zu verwendenden Metadaten vorgestellt. Metadaten werden über Informationsquellen, Ressourcen und extrahierte Produktinformationen im Repository vorgehalten. Bei jedem Metadatum müssen folgende Fragen berücksichtigt werden:

- Bezieht sich das Metadatum auf eine Informationsquelle, Ressourcen oder extrahierte Produktinformation?
- Kann das Datum immer gewonnen werden?
- Kann es sicher gewonnen werden? (→ Uncertain Information oder Fakt)
- Ist Angabe scharf oder unscharf?
- Wird es im Repository vorgehalten oder nur zur Laufzeit berechnet?
- Granularität – wie genau ist das Datum zu gewinnen? (vgl. Abschnitt 2.1.1, 2.4.4)

Darüber hinaus werden Metadaten auch über abgeleitete Fakten gewonnen und verwaltet, sodass zusätzlich der Aspekt des Reasonings betrachtet werden muss. Es gilt zu klären:

- Wie werden Metadaten über Fakt A und Fakt B zusammengeführt, wenn aus A und B der Fakt C geschlussfolgert wird?
- Welche Implikationen hat dies auf die Auswertung der Metadaten?
- Tritt unter Umständen ein Informationsverlust in den aggregierten Metadaten auf oder wird die Speicherung ineffizient, da alle Metadaten mitgeführt werden?

Für das Reasoning sind entsprechende Ableitungsregeln zu definieren, dieser Schritt kann jedoch erst zu einem späteren Zeitpunkt im Projekt erfolgen, da er ein mit Metadaten angereichertes Repository voraussetzt.

2.4.2 Klassen von Metadaten

URISource enthält einen eindeutigen Uniform Ressource Identifier, welcher auf eine Datenquelle oder Ressource zeigt. Die Angabe der *URISource* ist scharf und darf niemals leer sein, da die Quelle immer ermittelt werden kann. Zudem ist das Metadatum auch sicher ermittelbar, da die Gewinnung der URI keiner Interpretation eines Extraktionsprozesses unterliegt.

Das Metadatum URISource kann z.B. vom Anwender verwendet werden, um das Quelldokument (Ressource) als Ganzes zu betrachten oder Informationen über den File-Server zu erlangen, auf dem das Primärdatum gespeichert wird.

Author Wie in Abschnitt 2.3 bereits erläutert, sind unter dem Autor sowohl Menschen zu verstehen, als auch technische Geräte wie z.B. Sensoren in einem Wireless Sensor Network. Der Fakt, dass jede Information einen Autor besitzt impliziert jedoch nicht, dass dieser auch immer ermittelt werden kann. Abbildung 3 zeigt die Autorenschaft von Daten und die einhergehenden Probleme. Zunächst muss ermittelt werden, ob der Autor eindeutig ermittelt werden kann. Dieser Schritt ist primär vom Typ der Ressource abhängig. Betrachtet man z.B.

eine Datenbank mit mehreren Hundert Benutzern, so kann meist nicht festgestellt werden, welcher Nutzer welchen Eintrag vorgenommen hat. Bei der Gewinnung des Metadatums hat man dementsprechend die Wahl, alle Nutzer als mögliche Autoren festzuhalten oder gar keinen Autor anzugeben. Können die Autoren eindeutig ermittelt werden, da sie z.B. gemeinsam eine wissenschaftliche Veröffentlichung verfasst haben, muss unterschieden werden, ob es sich um einen einzelnen Autor oder einen Hauptautor und mehrere Coautoren handelt. Sind mehrere Autoren beteiligt, so muss für Anfragen an das Repository aus dem Metadatum *Author* ersichtlich sein, dass die Autoren die (von Aletheia extrahierten) Informationen *gemeinsam* in einem Dokument verfasst haben und nicht jeder die Information einzeln in einem separaten Dokument verfasst hat. Es gilt ebenso zu beachten, dass zwar die Autoren des Dokuments eindeutig identifizierbar sind, nicht jedoch der jeweilige Autor einer im Dokument enthaltenen Information.

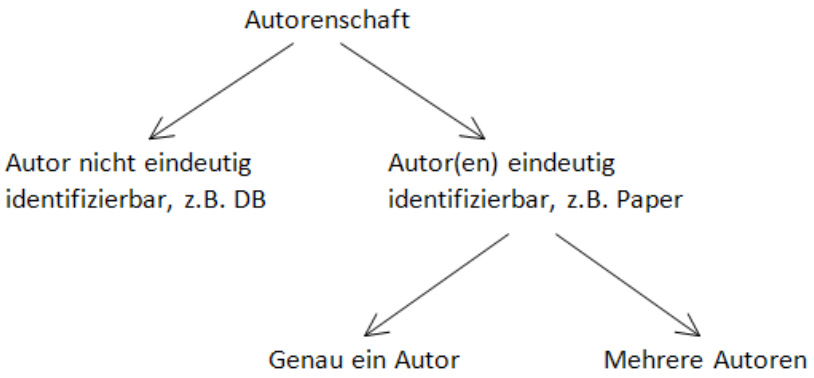

Abbildung 3: Autorenschaft von Daten

Da die Namen von Autoren nicht immer angegeben sind, sondern z.B. auch Forschergruppen als Autor eines Berichtes auftreten können, ohne die Angabe von Individuen, wird das Metadatum *Author* als unscharf betrachtet.

Die Angabe der Autoren kann in Ressourcen, speziell Dokumenten, auf unterschiedliche Weise erfolgen. In einem Forschungsbericht können die Autoren sowohl im Text selbst, als auch in dem dafür vorgesehenen Metadatenfeld angegeben sein. Der Vorteil der Extraktion der Autoren aus dem Metadatensatz des Dokumentes ist die Einfachheit, da bereits Tools hierfür existieren und kein zusätzliches Wissen über den Aufbau des Dokumenteninhaltes nötig ist. Der schwerwiegende Nachteil besteht in der Generierung der Metadaten durch die Autoren – nur die wenigsten fügen ihren Dokumenten diese Informationen hinzu. Wird eine Dokumentenvorlage verwendet und weitergegeben, so wird auch hier der Autor der Vorlage nur selten gegen die Autoren des Dokuments ersetzt, womit der Autor der Vorlage der vermeintliche Autor aller aus ihr entstandenen Dokumente ist. Das Aletheia-System kann dieses Manko nicht lösen, da es nicht dafür konzipiert ist, die Korrektheit der Ressource (und ihrer enthaltenen Metadaten) zu verifizieren. Es gilt festzuhalten, dass aus Metadaten extrahierte Autoren als sicher im Sinne der Extraktion betrachtet werden, wenngleich vorgenannte Probleme bei der Auswertung betrachtet werden müssen.

Das Metadatum *Author* kann unter anderem genutzt werden, um aus dem Autor und dem Kontext des Dokuments die Reputation des Verfassers abzuleiten.

DateCreated, DateChanged enthalten das Datum der Erstellung respektive der letzten Änderung der Ressource, aus denen die mit den Metadaten annotierte Information extrahiert wurde. Es gilt zu beachten, dass sich beide Daten auf die Quelle beziehen und ohne zusätzliche Informationen keine Rückschlüsse erlauben, ob eine extrahierte Information bereits zum Zeitpunkt der Erstellung der Ressource vorhanden war, erst bei der letzten Änderung hinzugekommen ist oder zwischenzeitlich geändert wurde. Aus diesem Grund und da uns kein Anwendungsfall bekannt ist, in dem die Speicherung beider Metadaten einen Mehrwert bringt, werden sie zu *DateChanged* zusammengefasst. *DateChanged* ist initial das Datum der Erstellung des Primärdatums und wird bei einer Änderung auf das Datum der letzten Modifikation gesetzt.

Wenngleich das Datum meist genau (d.h. scharf) ist, so kann es in manchen Anwendungsfällen vorkommen, dass die Genauigkeit des extrahierten Zeitstempels nicht den Anforderungen des Anwenders genügt. Ob das Datum sicher gewonnen werden kann, hängt von der Datenquelle ab. Während es in einem Dateisystem in der Regel einfach ausgelesen werden kann, ist bei der Gewinnung von Produktinformationen aus einem Web-2.0-Forum eine Extraktion aus dem Posting nötig, die fehlerbehaftet sein kann. Da das Metadatum nicht immer verfügbar ist, ist die Angabe im Repository fakultativ.

DateCrawling ist das Datum des Zugriffs des Data oder Information Providers auf die Ressource in der Datenquelle. Das Datum kann immer exakt, d.h. sicher und scharf, bestimmt werden und wird immer im Repository vorgehalten.

SourceDynamics wurde bereits in D.G3.2 [RWU09] vorgestellt und ist eine Einschätzung des Data oder Information Providers über die Dynamik der Quelle (z.B. File-Server, Forum), welche die Werte statisch, dynamisch oder hochgradig dynamisch einnehmen kann. Die Bewertung ist somit unscharf. Das Metadatum kann immer bestimmt werden und muss daher stets im Repository vorgehalten werden. Die Bestimmung selbst ist unsicher, da es nur eine Schätzung des Data oder Information Providers ist. Vor allem bei neu aufgenommenen Datenquellen kann sich das Verhalten der Quelle über die Zeit ändern, sodass nicht gewährleistet werden kann, dass die Einschätzung stets aktuell ist. Bei Bedarf kann das Metadatum zusätzlich über einzelne Dokumente in der Quelle vorgehalten werden, je nachdem, welche Granularität gefordert ist.

Das Metadatum *SourceDynamics* kann zum einen vom Data oder Information Provider verwendet werden, um entsprechend dieser Klassifikation unterschiedlich frequente Monitoring-Strategien einzusetzen, welche die Ressourcen auf Änderungen zu überprüfen. Zum anderen kann *SourceDynamics* mit *DateCrawling* und *DateChanged* aggregiert werden, um die Aktualität (Frische) einer Information im Repository zu bestimmen. Liegen z.B. *DateChanged* und *DateCrawling* dicht bei einander, jedoch weit in der Vergangenheit bezüglich des aktuellen Datums und ist die Quelle hochgradig dynamisch, so kann davon ausgegangen werden, dass die extrahierte Produktinformation nicht mehr aktuell ist. Liegt *DateChanged* hingegen weit in der Vergangenheit und *DateCrawling* sowie das Datum der Anfrage des Nutzers nahe bei einander, so kann er die Erkenntnis ziehen, dass die Information im Repository aktuell ist und dass sie sich auch seit *DateChanged* nicht mehr geändert hat.

Confidence ist ein Maß für die Qualität der Informationsextraktion einer Information aus einer Ressource. Sie wird benötigt, da Techniken wie Named-Entity-Recognition oder

Natural Language Processing zur Extraktion von Informationen aus semi- und unstrukturierten Quellen mit Unsicherheiten behaftet sind. Das Metadatum bezieht sich ausschließlich auf extrahierte Informationen und niemals auf Datenquellen oder Ressourcen. Es ist scharf, kann jedoch nicht sicher angegeben werden, da es eine Einschätzung der Extraktionskomponente ist, wie sicher sich diese ist, dass die extrahierte Information auch tatsächlich so in der Ressource enthalten ist. Die *Confidence* muss zu jeder Information angegeben werden. Aussagen mit einer *Confidence* von 1.0 werden als Fakten bezeichnet, alle anderen als unsichere Informationen.

Die *Confidence* wird von mehreren Komponenten ausgewertet. Anhand der Unterscheidung zwischen Fakten und unsicheren Aussagen wird entschieden, in welchem Repository eine Information vorgehalten wird und ob sie dem Reasoning zur Verfügung steht. Beim Ranking von Suchergebnissen fließt sie in die Bewertung der Relevanz ein. Je nach Wunsch der Anwendungspartner kann das Metadatum dem Suchenden mit präsentiert werden.

RelevanceScore ist ein Maß für die Relevanz eines gesuchten Terms innerhalb der Ergebnismenge. Er wird z.B. mittels TF-IDF[7] zur Anfragezeit bestimmt und ist daher von einer Query und den in diesem Moment im Repository vorhandenen Informationen abhängig, sodass das Metadatum *RelevanceScore* nicht im Repository vorgehalten wird. Die Berechnung wird sowohl als scharf, als auch als sicher angesehen. Die Gewinnung ist immer möglich, sodass der Wert immer angegeben werden muss. Der *RelevanceScore* wird zur Sortierung der Ergebnisse einer Anfrage verwendet.

DateValidFrom, **DateValidUntil** geben Auskunft über die Gültigkeitsdauer von Informationen. Eine begrenzte Gültigkeitsdauer liegt z.B. bei Aktionsangeboten oder (zeitlich begrenzten) Verträgen vor. Diese Metadaten liegen nicht bei allen Ressourcen vor und können daher nicht immer gewonnen werden. Aufgrund der Extraktion sind sie als unsicher zu betrachten, zudem können sie unscharf sein.

DateValidFrom und *DateValidUntil* können in die Bewertung der Relevanz einfließen; sucht ein Nutzer z.B. nach seinen aktuellen Vertragsbedingungen, so werden die AGBs abgelaufener Verträge nicht mit als Ergebnis zurück gegeben.

Die Liste der vorgenannten in Aletheia zu betrachtenden Qualitätsmetadaten ist nicht abgeschlossen, sodass sie im weiteren Projektverlauf bei Bedarf problemlos erweitert werden kann. Die Anforderung der Beschränkung der Zugriffsrechte auf Produktinformationen sowie die sie beschreibenden Qualitätsmetadaten wird gemeinsam mit dem Arbeitspaket AP.G7 abgestimmt.

2.4.3 Klassifikation verwendeter Qualitätsmetadaten

Tabelle 3 fasst die nach aktuellem Kenntnisstand in Aletheia benötigten Qualitätsmetadaten und ihre Eigenschaften noch einmal übersichtlich zusammen. Eine Klassifikation kann anhand jeder Spalte vollzogen werden, je nachdem, aus welchem Blickwinkel die Qualitätsmetadaten betrachtet werden. Soll z.B. zwischen im Repository vorgehaltenen und zur Laufzeit gewonnenen Qualitätsmetadaten unterschieden werden, werden, so wird die

[7] TF-IDF: term frequency - inverse document frequency, siehe [BYRN99]

vorletzte Spalte betrachtet. Bis auf *RelevanceScore* können alle Qualitätsmetadaten vorab berechnet und in Repository vorgehalten werden.

Qualitäts-metadatum	Qualitäts-metadatum bezieht sich auf	Gewinnung immer möglich	Darf leer sein	Datum immer Scharf	Gewinnung immer sicher	Im Repository vorgehalten	Quelle des Qualitäts-metadatums
URISource	D, R	Ja	Nein	Ja	Ja	Ja	R
Author	R	Nein	Ja	Nein	Nein	Ja	A, R
DateChanged	R	Nein	Ja	Nein	Nein	Ja	R
DateCrawling	R	Ja	Nein	Ja	Ja	Ja	A
SourceDynamics	D, R	Ja	Nein	Nein	Nein	Ja	A
Confidence	I	Ja	Nein	Ja	Nein	Ja	A
RelevanceScore	R, I	Ja	Nein	Ja	Ja	Nein	A
DateValidFrom	R	Nein	Ja	Nein	Nein	Ja	R
DateValidUntil	R	Nein	Ja	Nein	Nein	Ja	R
Legende:	A = Aletheia, z.B. Extraktionskomponente; D = Datenquelle, z.B. File-Server; I = extrahierte Information, z.B. “Canon50D hatSensorTyp APS-C“; R = Ressource, z.B. Office-Dokument						

Tabelle 3 Qualitätsmetadaten in Aletheia

2.4.4 Granularität der Qualitätsmetadaten

Wie bereits in Abschnitt 2.4.2 ersichtlich wurde, können sich Qualitätsmetadaten auf verschiedene Ebenen wie Informationsquellen, Ressourcen und extrahierte Produktinformationen beziehen. Betrachtet man zum Beispiel die Vertrauenswürdigkeit einer Quelle, dann gibt es einen signifikanten Unterschied, ob die Vertrauenswürdigkeit einer konkreten extrahierten Information bewertet wird, oder die der Informationsquelle als Ganzes. Dies sei an einem Beispiel veranschaulicht:

Im Web 2.0 veröffentlicht eine rasant wachsende Anzahl von Autoren ihre Erfahrungen und Meinung zu Produkten, wobei sowohl ihr Wissen, als auch ihre Motivation sehr unterschiedlich sind. Ein Vorteil fein granularer Qualitätsmetadaten ist die Möglichkeit, von Experten veröffentlichte Informationen anders zu gewichten als die von Anderen. Ein Nachteil ist der erhöhte Aufwand für die Bewertung jeder einzelnen Information. Die Bewertung einer Informationsquelle als Ganzes schmälert den Aufwand, resultiert jedoch in dem Nachteil, dass alle aus der Quelle extrahierten Informationen mit der gleichen, durchschnittlichen Bewertung versehen werden. Weder das föderierte Produktinformationssystem, noch dessen Nutzer können zwischen Informationen von Experten und von Laien unterscheiden.

Im weiteren Projektverlauf gilt es zu untersuchen a) welche Stufen der Granularität sich generell identifizieren lassen, b) welche Stufen bei welcher Informationsquelle auftreten (z.B. in Web 2.0 Forum: komplettes Forum, einzelner Thread, einzelner Beitrag), c) inwiefern die Granularität vom speziellen Typ der Qualitätsmetadaten und dem Typ der Informationsquelle oder Ressource abhängig sind bzw. welche weiteren Abhängigkeiten sich identifizieren lassen.

3 Methoden zur Gewinnung von Qualitätsmetadaten

Das vorherige Kapitel gab einen Überblick über die Kategorisierung und Identifkation relevanter Qualitätsmetadaten, welche sowohl für interne Verarbeitungsprozesse als auch für den Endnutzer von Aletheia wichtige Zusatzinformationen bei produktbezogenen Suchen darstellen. In diesem Kapitel werden Methoden vorgestellt, die in Aletheia zur weitestgehend automatisierten Gewinnung von Qualitätsmetadaten genutzt werden können. Dabei wird ausgehend von der Architektur der Datenintegrationsplattform ein allgemeiner und wiederverwendbarer Prozess zur Gewinnung von Qualitätsmetadaten definiert. Es wird aufgezeigt, wie dieser Prozess in die bestehenden Abläufe der Informationsgewinnung in Arbeitspaket AP.G3 integriert werden kann und welche Komponenten hierfür zuständig sind.

3.1 Architekturüberblick der Datenintegrationsplattform

Die Gesamtarchitektur der Datenintegrationsplattform ist in [RWU09] beschrieben. Abbildung 4 zeigt den für dieses Deliverable relevanten Teilbereich, der auch für die Gewinnung von Qualitätsmetadaten von Relevanz ist.

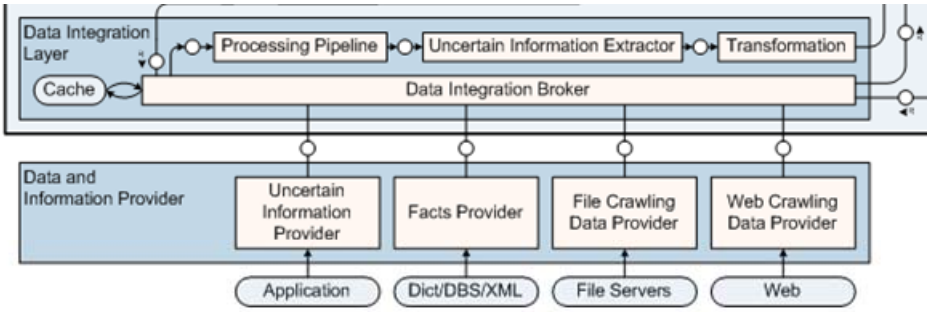

Abbildung 4: Datenintegration in der Aletheia Architektur

3.2 Allgemeiner Prozess der Gewinnung von Qualitätsmetadaten

Die Aletheia-Verarbeitungs-Pipeline für unstrukturierte und semistrukturierte Datenquellen bildet eine zentrale Komponente in der Architektur der Datenintegrationsplattform, die eine komplexe Strategie zur Verarbeitung von heterogenen Dokumenten darstellt. Sie besteht u.a. aus den in Abbildung 5 dargestellten Transformationsschritten.

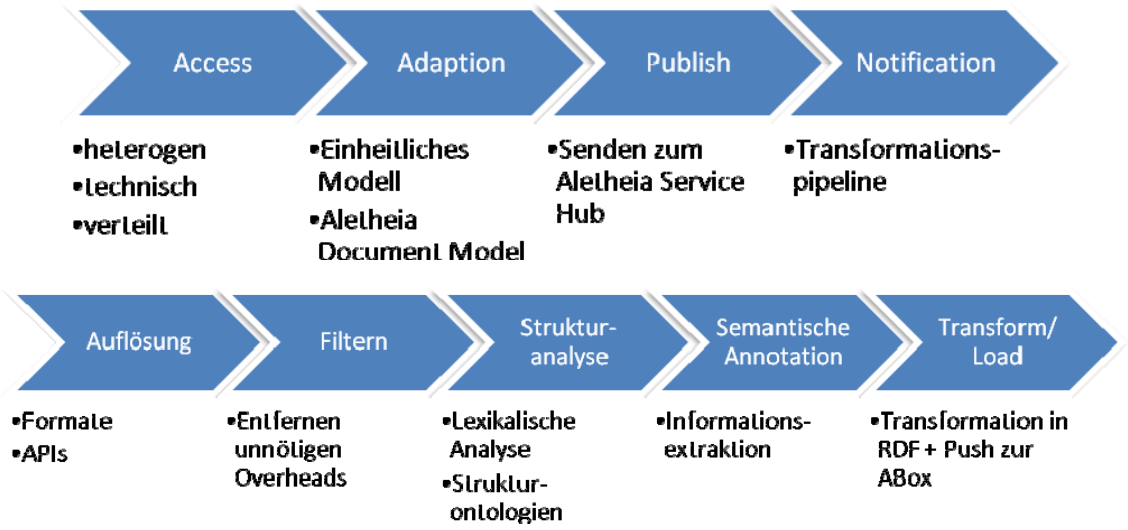

Abbildung 5: Verarbeitungspipeline für Aletheia Dokumente

Jeder dieser Schritte (vgl. [RWU09]) erzeugt Metadaten. Nur eine Teilmenge dieser Information ist dabei auch wirklich relevant in Bezug auf die Qualität der gewonnenen Informationen. Tabelle 4 stellt mögliche Beispiele für Qualitätsmetadaten dem entsprechenden Verarbeitungsschritt gegenüber.

Schritt	Qualitätsmetadatum
Access	Latenzzeit des Zugriffs
Adaption	-
Publish	Integrationszeitpunkt der Information
Notification	-
Auflösung	-
Filtern	Informationsdichte innerhalb eines Dokuments
Strukturanalyse	Datenkomplexität
Annotation	Konfidenz der semantischen Annotation
Transformation	-

Tabelle 4: Beispiele für Qualitätsmetadaten in Bezug zu deren Erzeugungszeitpunkt

Qualitätsmetadaten werden also im Laufe der semantischen Integration verschiedener heterogener Datenquellen an die zugehörigen Verarbeitungsartefakte angehängt. Der Begriff „Verarbeitungsartefakt“ steht dabei für:

- **Aletheia-Dokumente** (vgl. [RWU09]), wenn man unstrukturierte oder semistrukturierte Datenquellen betrachtet,

- **semantische Aussagen**, also Subjekt-Prädikat-Objekt-Tripel bzw. Fakten[8], betrachtet man strukturierte Datenquellen.

An dieser Stelle sei erneut auf Deliverable D.G3.2 [RWU09] verwiesen. Dort wird das Modell des Aletheia-Dokuments beschrieben, das als vereinheitlichte Schnittstelle zur Verarbeitung der unstrukturierten und semistrukturierten Daten dient. Neben der semantischen Annotation und dem entsprechenden Inhaltsbereichen enthält es auch Metadaten verschiedenen Typs. Es ist unter anderem erweiterbar auf Metadatentypen, die sich speziell auf die Qualität beziehen.

3.3 Methoden zur Gewinnung konkreter Qualitätsmetadaten

In diesem Abschnitt werden die in Kapitel 2 erläuterten Qualitätsmetadaten wieder aufgegriffen. Für jedes Datum wird ein Beispielszenario erläutert, welches die Relevanz des Qualitätsmetadatums im Hinblick auf die Suchfunktionalität innerhalb von Aletheia herausstellt. Gleichzeitig werden entsprechende Methoden beschrieben, wie diese Qualitätsmetadaten automatisch, semiautomatisch oder manuell gewonnen werden können.

Metadatum	**Kurzbeschreibung**	**Methode der Gewinnung**
URISource	Während des Suchvorgangs nach Informationen kann eine Anforderung darin bestehen, bestimmten Quelltypen oder Data- bzw. DocumentProvider zu filtern oder auszuschließen. Das Metadatum *URISource* ermöglicht dies.	URIs von Informationsquellen werden einerseits durch Data- und DocumentProvider generiert, andererseits innerhalb des Service-Hubs erzeugt. Dabei liegen dieser Generierung entsprechende Konfigurationen zugrunde. URIs lassen sich aus anderen Metadaten, wie Lokation, Version und Timestamp erzeugen, welche wiederum den Data- und Information Providern zur Verfügung stehen.
Author	Die Autorenschaft von Informationen kann während einer Suche gezielt genutzt werden, um implizite Qualitätsinformation von Dokumenten und der darin enthaltenen Informationen zu gewinnen. Autoren können aufgrund einer manuellen Bewertung ihrer früheren Beiträge als Klassifkations-	Das Merkmal „Autor“ einer Informationsquelle kann nur begrenzt automatisch gewonnen werden. In den meisten Datenquellen (z.B. Blogs, Foren, Content Management Systeme) ist ein Metadatum für Autor generell vorhanden und kann einfach ausgelesen werden. Andere Dokumente führen dieses Qualitätsmetadatum inhärent mit

[8] Während Fakten immer als wahr angenommen werden, können Aussagen falsch oder auch unsicher sein.

Metadatum	Kurzbeschreibung	Methode der Gewinnung
	merkmal für die Qualität der Datenquelle dienen.	sich. In diesem Fall können diese Metadaten über Frameworks zur Metadatenextraktion (Apache Tika[9], Aperture[10]) gewonnen werden.
DateChanged	Ein sehr wichtiges Qualitäts-metadatum ist die Aktualtiät einer Datenquelle. DataChanged beschreibt die letzte Änderung an einem Dokument und dient damit als Filtermerkmal für Dokumente. Beispielsweise können so Dokumente, die älter als ein Jahr sind, von einer Suche ausgenommen werden.	Dieses Merkmal lässt sich gewöhnlich über Standardmecha-nismen (vgl. Apache Tika, Aperture) gewinnen, da es meist Bestandteil des im Dokument enthaltenen Meta-datensatzes ist. Anderfalls kann es entweder über die Dokumenten-umgebung (Fileshares) oder über Zeitausdrücke in Texten (z.B. in Foren) gewonnen werden. Letztere werden mittels entsprechender DocumentProvider bereitgestellt.
SourceDynamics	Die Dynamik einer Datenquelle kann ein Hinweis auf die Aktualität der Information sein.	Die Klassifikation einer Datenquelle in ihre Dynamikklasse wird von Document Providern vorgenommen. Diese überwachen die Änderungs-frequenz von Datenquellen.
DateCrawling	Vergleichbar zu *DateChanged* beschreibt dieses Metadatum, zu welchem Zeitpunkt auf die extern vorliegenden Datenquellen zugegriffen wurde.	Entsprechende DocumentProvider und Crawling-Komponenten hängen dieses Metadatum zum entsprechenden Crawl-Zeitpunkt an ein Dokument an.
Confidence	Die *Confidence* ist eine Selbsteinschätzung einer Extraktionskomponente über die Genauigkeit eines Extraktionsergebnisses, welche bedingt durch Mehrdeutigkeiten und auftretende Fehler schwanken kann.	Das Metadatum wird von der jeweiligen Extraktionskomponente anhand des zu erkennenden Datensatzes generiert. Dieser bewertende Prozess kann je nach Extraktionsmethodik unter-schiedlich komplex sein.
RelevanceScore	Der *RelevanceScore* beschreibt die Relevanz von Termen in Dokumenten. Dieses Metadatum wird implizit bei jeder Suchanfrage abgefragt. Jeder Term wird in Verbindung	Der *RelevanceScore* bezieht sich auf alle Terme aller Dokumente und wird offline berechnet (z.B. TF-IDF Score).

[9] http://lucene.apache.org/tika/
[10] http://aperture.sourceforge.net/

Metadatum	Kurzbeschreibung	Methode der Gewinnung
	mit den Termen der Suchanfrage gebracht. Die Ergebnisliste kann dann entsprechend sortiert und gefiltert werden.	
DateValidFrom *DateValidUntil*	Beschreibt den Zeitpraum, in dem eine Information gültig ist. Ein Beispiel hierfür ist ein zeitlich begrenztes Sonder-angebot eines Produkts.	Speziell trainierte Algorithmen zur Informationsextraktion können in Sonderfällen Aussagen über die Gültigkeit von Informationen aus Dokumenten extrahieren.

3.4 Erweiterung der Datenintegrationsplattform

Betrachtet man die in Abbildung 5 dargestellte Verarbeitungspipeline für Dokumente und Daten innerhalb von Aletheia, so kann man diese auch als Pipeline zur Metadatenanreicherung sehen. Jeder einzelne Schritt in der Verarbeitungsreihenfolge ergänzt dabei den entsprechenden Datensatz um Metadaten, welche schließlich zusammen mit den Extraktionsergebnissen gespeichert werden (vgl. Kapitel 4). Für die Metadatenanreicherung werden also keine zusätzlichen Komponenten benötigt, vielmehr wird entsprechende Funktionalität an den verschiedenen Verarbeitungspunkten in den folgenden Komponenten umgesetzt:

- Data and Information Provider Layer

  Die Komponenten dieses Layers, welche dem Zugriff auf Daten dienen, ergänzen zu den zu beziehenden Daten, Qualitätsmetadaten wie: *URISource*, *DateChanged*, *SourceDynamics*, da sie vorkonfiguriert über das entsprechende notwendige Wissen wie Lokation, Zeitpunkt und Dynamikverhalten der Quelle besitzen.

- Data Integration Broker

  Der Data Integration Broker wird eingehende Daten und Information mit weiteren Qualitätsmetadaten (z.B. *DateCrawling*) versehen.

- Uncertain Information Provider

  Diese Komponenten dienen neben der Extraktion von primären Informationen aus Ressourcen auch der Gewinnung von Qualitätsmetadaten. Hierbei werden verschiedene Komponenten folgende Qualitätsmetadaten extrahieren bzw. bei der Extraktion generieren: *Author*, *RelevanceScore*, *Confidence*, *DateValidFrom*, *DateValidUntil*

Neben den genannten Stellen zur Integration von Qualitätsmetadaten bleibt die Datenintegrationsplattform flexibel in deren Handhabung. Während der Verarbeitung der

Daten und Dokumente, werden Qualitätsmetadaten im in Deliverable D.G3.2 beschriebenen Dokumentenmodell gehalten. Dieses ermöglicht es jeder Komponente, diese Qualitätsmetadaten zu lesen und zu verändern.

4 Methoden zur Verwaltung von Qualitätsmetadaten

Dieses Kapitel befasst sich mit der Verwaltung von Qualitätsmetadaten. Diese sollen in Aletheia einheitlich mit den übrigen Metadaten verwaltet werden. Sowohl die Gewinnung, die Speicherung als auch der Zugriff auf die Daten zählen zu Aspekten der Verwaltung. Die Art und Weise der Datenspeicherung wirkt sich signifikant darauf aus, zu welchem Zeitpunkt Metadaten generiert werden müssen und auf die Zugriffsgeschwindigkeit. Aus diesem Grund wird auf die Gesichtspunkte, die sich mit der Speicherung befassen, in diesem Kapitel das Hauptaugenmerk gelegt.

4.1 Grundlagen und State-of-the-Art bei der Verwaltung von Metadaten

Dieser Abschnitt befasst sich mit Grundlagen und Konzepte der RDF-Speicherung, die für das Verständnis der folgenden Abschnitte bzw. Kapitel notwendig sind. Im Anschluss daran werden Ansätze zur Verwaltung von Metadaten genauer erläutert. Die Verwaltung von Metadaten ist aktueller Gegenstand der Forschung. Daher handelt es sich bei den in Kapitel 4.2.2 und 4.2.3 vorgestellten Ansätzen um experimentelle Ansätze die noch nicht über Produktreife verfügen.

4.1.1 RDF-Grundlagen

In [DHB+09] wurden die wichtigsten Ontologiesprachen ausführlich beschrieben. An dieser Stelle erfolgen daher nur eine knappe Wiederholung von RDF, sowie die Beschreibung von spezifischen, für die Metadatenverwaltung notwendigen Formalismen/Methoden, die [DHB+09] teilweise entnommen wurden.

RDF (Resource Description Framework) stellt die unterste semantische Schicht im Semantic Web Technology Stack dar.[11] Dabei ist die Kernidee von RDF, Aussagen über Ressourcen treffen zu können, wobei die Aussagen Tripel der Form `(Subjekt Prädikat Objekt)` sind. Eine RDF-Wissensbasis ist im Wesentlichen eine Ansammlung derartiger Tripel.

Ressourcen werden in RDF über URIs (Uniform Resource Identifiers), die als eine Verallgemeinerung von URLs verstanden werden können, identifiziert. Neben URIs können die Elemente eines Tripels auch Literale oder leere Knoten (englisch: blank nodes), die anonyme oder unbekannte Ressourcen repräsentieren, sein. Für die Literale kann man in RDF auf die Datentypen von XML Schema zurückgreifen.

Reifikation[12]

Durch Reifikation ermöglicht RDF, dass Aussagen über Aussagen getroffen werden können. Reifikation bezeichnet im wörtlichen Sinne die Vergegenständlichung. Das bedeutet, dass aus einem Ereignis ein Objekt gemacht wird. Um dies in RDF zu realisieren, müssen die Aussagen, über die Aussagen getroffen werden sollen, zu einer Ressource zusammengefasst werden. Diese zusammenfassende Ressource wird im Anschluss nochmals mittels RDF-beschrieben. Der umschriebene Prozess wird mit Reifikation bezeichnet. Das resultierende Modell einer Aussage wird Reified Statement genannt. RDF definiert folgende Eigenschaften um Aussagen zu modellieren:

[11] http://www.w3.org/2007/Talks/0130-sb-W3CTechSemWeb/#(24)
[12] http://www.ai.wu.ac.at/usr/albert/node32.html

- Subjekt: Die Eigenschaft rdf:subject definiert das Subjekt der zu modellierenden Aussage.
- Prädikat: Die Eigenschaft rdf:predicate gibt das Prädikat wieder, das dem Subjekt in der ursprünglichen Aussage zugeteilt ist.
- Objekt: Die Eigenschaft rdf:object definiert das Objekt, das dem Subjekt in der Aussage zugewiesen wird.
- Typ: Die Eigenschaft Typ spezifiziert den Typ der neu entstandenen Ressource. Alle Reified Statements sind Instanzen von rdf:Statement. Das bedeutet, dass die Aussage RDF:Statement oder eine Klasse die RDF:Statement integriert, als Typenbezeichnung zugeordnet werden muss.

Auf diese Weise entsteht eine neue Ressource, die mit mindestens vier Eigenschaften verknüpft ist, und die als Subjekt oder Objekt in weiteren Aussagen verwendet werden kann. Werden viele Aussagen reifiziert, kann dies zu einem triple bloat führen, das heißt zu einer Vervielfachung der in der Datenbasis gehaltenen Tripel. Das Konzept der RDF-Reifikation wird im Folgenden an einem Beispiel illustriert. Ein einfaches, nicht reifiziertes Tripel mit der Aussage `Die Kamera K wird von Hersteller H produziert.` sieht in RDF und als Tripel folgendermaßen aus:

```
<rdf:Description rdf:about="K">
    <hasProducer rdf:resource="H"/>
</rdf:Description>
```

```
[:K   :hasProducer      :H]
```

Die Aussage `PersonX schreibt, dass Kamera K von Hersteller H produziert wird.` wird wie in Abbildung 6 dargestellt.

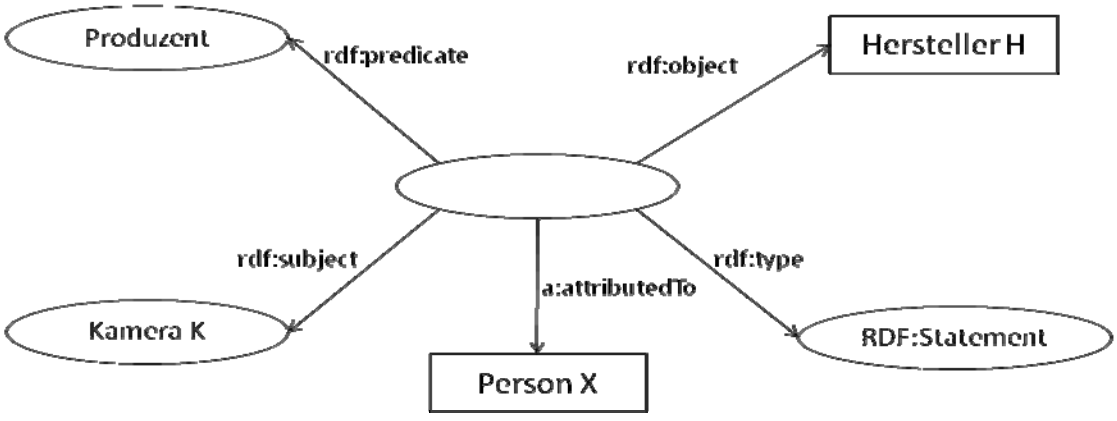

Abbildung 6: Beispielschema für reifizierte Aussage

Die Abbildung zeigt, wie einerseits das Subjekt (Kamera K), das Prädikat (Produzent), das Objekt (Hersteller H) und der Typ (rdf:Statement) dargestellt werden. Zusätzlich wird über a:attributedTo eine Angabe über die Person repräsentiert, die diese Aussage getroffen hat.

Nach der RDF-Reifikation sehen das RDF und die Tripel wie folgt aus:

```
<rdf:Statement a:attributedTo="PersonX">
      <rdf:subject rdf ressource="K"/>
      <rdf:predicate rdf:resource="&;Produzent"/>
      <rdf:object>H</rdf:Object>
</rdf:Statement>


[:triple123   rdf:type        rdf:Statement]
[:triple123   rdf:subject     :K]
[:triple123   rdf:predicate   :hasProducer]
[:triple123   rdf:object      :H]
[:triple123   dc:creator      :PersonX]
```

Quadrupel

Quadrupel, auch Quads genannt, erweitern RDF-Aussagen um eine vierte Komponente. Das RDF-Tripel <S, P, O> (Subjekt, Prädikat, Objekt) wird somit zu <C, S, P, O>. Dabei dient die zusätzliche Komponente C der Kontextbildung und wird zur Gruppierung der Aussagen genutzt. C kann mit *null* belegt sein oder eine Ressource angeben. Sollen Aussagen über eine Menge/Gruppe von Aussagen konstatiert werden, kann die Kontextkomponente C auch als Subjekt oder Objekt des Tripels verwendet werden. Das obige Beispiel wird nun um zwei Informationen erweitert: PersonX schreibt zusätzlich, dass `Die Kamera K ist am 12.05.2009 erschienen`. Diese Information veröffentlicht sie auf der Webseite `www.example.com/`. Das modifizierte Beispiel hat folgende Quadrupel-Notation und orientiert sich an der Notation von [Tol06]:

```
[_:id123 _:K      :hasProducer      _:H]
[_:id123 _:K      :hasReleasedate   "12.05.2009"]
[null    _:id123  dc:creator        _:PersonX]
[null    _:id123  :hasSource        "www.example.com"]
```

Named Graphs

Eine Weiterentwicklung von Quadrupeln sind Named Graphs. Hier werden Graphen gebildet, die ebenfalls Gruppierungen von Aussagen darstellen. Ein Graph wird durch einen Namen gekennzeichnet. Dieser kann eine URI oder eine anonyme Ressource sein und wieder in der RDF-Aussage verwendet werden. Zur Überführung von Quadrupeln in die Named Graphs Darstellung kommen die TriG oder die TriX Syntax zum Einsatz, auf die im folgenden Abschnitt näher eingegangen wird. Das fortlaufende Beispiel lautet in der TriG Syntax:

```
_:G1  (_:K        :hasProducer      _:H.
       _:K        :hasReleasedate   "12.05.2009".)
_:G2  (_:G1       dc:creator        _:PersonX.
       _:G1       :hasSource        "www.example.com".)
```

RDF-Quadrupel-Serialisierungen

TriX[13] (RDF-Tripels in XML) ist eine experimentelle alternative Serialisierung von RDF-Tripeln in XML, die von Jeremy Caroll (HP Labs) und Patrick Stickler (Nokia) entwickelt wurde. Sie wurde entwickelt um Probleme der existierenden Serialisierung RDF/XML zu beheben. Zu den Problemen gehören beispielsweise, dass in XHTML und anderen XML-Dokumenten eingebettetes RDF sehr schwer zu validieren ist. Weiterhin ist es nicht möglich, eine Untermenge von RDF/XML zu definieren, die alle RDF-Graphen repräsentieren und durch eine DTD oder ein XML Schema beschrieben werden kann. Auf der Webseite wird die TriX Syntax folgendermaßen beschrieben:

Das Tripel-Element enthält drei Kinder (Subjekt, Prädikat, Objekt) und bildet den Kern der TriX-Syntax. Jedes dieser Kinder kann ein URI-Element, ein ID-Element, ein plainLiteral oder ein typedLiteral Element sein. Die Ausprägung hängt davon ab, ob der entsprechende Knoten des Graphs eine RDF-URI-Referenz, ein Blank Node oder ein Literal ist. Der Inhalt eines Elements besteht aus dem Label des Knoten, beziehungsweise dem Identifikator des Blank Node. Named Graphs werden in TriX durch die Verwendung eines optionalen URI-Elements vor dem Tripel des Graphs unterstützt.

TriG[14] ist ein plain text Format für die Serialisierung von Named Graphs und RDF-Datensätzen, das eine kompakte und lesbare Alternative zur XML-basierten TriX Syntax darstellt. TriG ähnelt Turtle (Terse RDF-Triple Language[15]), besitzt jedoch folgende Erweiterungen:

- die Symbole ‘{ ‘und ‘}‘, um Tripel in verschiedene Graphen zu gruppieren und um Named Graphs über ihre Namen anzusprechen.
- optionaler ‘=‘ Graph naming Operator und optionaler ‘.‘ nach jedem Graph für N3 Kompatibilität.

Zudem gelten zwei zusätzliche Regeln. Graph Namen müssen innerhalb eines TriG Dokuments einheitlich sein. Es muss mindestens einen unbenannten Graph in einem TriG Dokument geben.

Abhängig von der Gewinnung der Metadaten ist im Folgenden zwischen sicheren und unsicheren Metadaten zu unterscheiden. Metadaten wie *URISource* oder *DateCrawling* sind sichere Metadaten. Diese können in einem semantischen Repository verwaltet werden. Wird zum Beispiel das Datum der letzten Änderung einer Ressource aus dem Dateisystem gelesen, so handelt es sich auch hier um sichere Metadaten (vgl. 2.4.2 Definition *DateChanged*). Durch den Uncertain Information Provider extrahierte Informationen stellen unsichere Metadaten dar. Hierbei handelt es sich beispielsweise um aus dem Text eines Forums extrahierte Angabe der letzten Änderung, aber auch Metadaten wie *RelevanceScore* oder *Confidence*. Diese Metadaten können nicht in einem semantischen Repository verwaltet werden. Möglichkeiten zur Handhabung der unsicheren Metadaten werden in Kapitel 4.1.3 und 4.3.2 erläutert.

[13] http://sw.nokia.com/trix/TriX.html
[14] http://www4.wiwiss.fu-berlin.de/bizer/TriG/Spec/
[15] http://www.w3.org/TeamSubmission/turtle/

4.1.2 Verwaltung sicherer Metadaten

MetaK

In [DSSS09] wird ein Ansatz zur Repräsentation, Speicherung und Abfrage von Metadaten in RDF vorgestellt. Konkret geht es dabei um Metadaten wie Herkunft, Anbieter, Erstellungsdatum und Glaubwürdigkeit. Nachfolgend werden die im Ansatz von Dividino et al. getroffenen Entwurfsentscheidungen erläutert.

Sollen Verknüpfungen zwischen Wissen und Metawissen hergestellt werden, setzt dies die Existenz geeigneter Reifikationsmechanismen voraus. Um die Kompatibilität mit bereits existierenden Anwendungen zu gewährleisten, dürfen die ursprünglichen Nutzerdaten nicht verändert werden. Für den vorgestellten Ansatz entfällt daher die Möglichkeit der RDF-Reifikation, da hierbei das Repräsentationsmodell verändert wird, wie das Beispiel in Kapitel 4.1.1 zeigte. Damit die Tripel nicht zerlegt werden, verwenden Dividino et al. Named Graphs. Entsprechend der RDF-Philosophie trennen die Autoren nicht zwischen Wissen (Nutzerwissen) und Metawissen. So kann ein Nutzer auch direkt auf das Metawissen zugreifen.

Die Autoren führen eine abstrakte Syntax für RDF$^+$ ein, die auf den Bausteinen von RDF aufbaut: URIs, RDF-Literale, eine Menge von Graph Namen und eine Menge von Properties. Zusätzlich definieren sie eine Menge von Aussagenidentifikatoren, die disjunkt von den zuvor festgelegten Mengen der URIs und RDF-Literalen ist. Auf diese Weise können RDF$^+$ Literalaussagen ausgedrückt werden, die in einen Named Graph eingebunden sind und einen globalen eindeutigen Identifikator besitzen. In [DSSS09] wird ein RDF$^+$ Meta Knowledge Statement folgendermaßen definiert:

Sei $\Pi \subseteq P$ die Menge der Meta-Knowledge-Eigenschaften, dabei ist P die Menge aller Eigenschaften. Sei Ω_π mit $\pi \in \Pi$ die Menge, die mögliche Wertebereiche der Meta-Knowledge-Eigenschaften $\pi \in \Pi$ enthält. Dann ist die Menge $\mathfrak{M}$ aller RDF$^+$ Meta Knowledge Statements definiert durch:

$$\mathfrak{M} := \{(\theta, \pi, \omega) | \theta \in \Theta, \pi \in \Pi, \omega \in \Omega\}$$

Dabei ist Θ eine Menge von Statement-Identifikatoren, die disjunkt zu den Mengen der URIs und Literale ist. Dies sei anhand eines Beispiels aus [DSSS09] verdeutlicht. Zu extrahierten Daten liegen folgenden Metadaten vor:

```
G3 {  G1 mk:source <www.rpi.edu/report.doc> .
      G1 mk:certainty "0.9" .}
```

Unter Verwendung der Definition für RDF$^+$ Meta Knowledge Statements werden diese Aussagen wie folgt dargestellt:

$\mathfrak{M} \supseteq M \supseteq$ {
( , mk:source, {<www.rpi.edu/report.doc>}),
( , mk:certainty, 0.9)}

Um RDF+ sinnvoll anfragen zu können, wird die SPARQL-Syntax um den Ausdruck „WITH META Metalist“ erweitert. Auf diese Weise können die in Metalist genannten Named Graphs in die Abfrage mit einbezogen werden. Die erweiterte SPARQL SELECT Anfrage ergibt sich zu:

```
SELECT SelectExpression
(WITH META MetaList)?
(FROM GraphName)+
WHERE P
```

Analog hat eine erweiterte SPARQL CONSTRUCT Anfrage folgende Form:

```
CONSTRUCT ConstructExpression
(WITH META MetaList)?
(FROM GraphName)+
WHERE P
```

MetaK ermöglicht unterschiedliche Erweiterungen. Denkbar sind hier etwa die Unterstützung von zusätzlichen Auswahl- oder Ranking-Bedingungen für assoziierte Metawissens-Attribute. Diese Funktionalität kann als Erweiterung der WHERE Klausel aufgefasst werden. Auch eine Erweiterung des Frameworks, so dass Filter- oder Ranking Bedingungen auf Metawissen und Query-Ergebnisse angewendet werden können, ist möglich. Diese Funktionalität kann konzeptionell als Erweiterung der HAVING-Klausel gesehen werden (im allgemeinen SQL-Sinn).

Coloring RDF-Triples to Capture Provenance

In [FFP+09] wird ein Ansatz zur Repräsentation der Herkunft von RDF-Daten und Schema Tripeln mittels Farben vorgestellt. Dabei repräsentiert die Farbe eines Tripels die Quelle, aus der es gewonnen wurde. Die Farbe eines RDF-Tripels wird als vierte Spalte aufgenommen, so dass man RDF-Quadrupel erhält.

Ein Hauptaspekt des Papers ist die Erweiterung von RDFS-Inferenzregeln, so dass die Herkunft impliziter RDFS-Tripel festgestellt werden kann. Von einem impliziten Tripel spricht man, wenn das Tripel nicht explizit angelegt, sondern durch Inferenzregeln, zum Beispiel Transitivität, abgeleitet wurde. Um die Herkunft eines impliziten Tripels zu repräsentieren wird im Paper folgende Semigruppe definiert:

Def: Die Struktur $(I, +)$ ist eine Semigruppe mit

- $I \subset U$ ist die Menge der Farben, die disjunkt von der Menge der Klassen C und der Propertytypen P ist.
- “+“ ist eine binäre Operation mit den folgenden Eigenschaften für alle $c_1, c_2, c_3 \in I$

$$c_1 + c_1 = c_1 \qquad \text{(Idempotenz)}$$

$$c_1 + c_2 = c_2 + c_1 \qquad \text{(Kommutativität)}$$

$$c_1 + (c_2 + c_3) = (c_1 + c_2) + c_3 \qquad \text{(Assoziativität)}$$

Die binäre Funktion"+" beschreibt die Herkunft impliziter RDF-Tripel und gibt eine Verbundherkunft zurück, die ebenfalls in $\boldsymbol{I}$ liegt. Das bedeutet, dass folgender Sachverhalt gilt: ein Tripel t wird impliziert durch ein Tripel t_1 (dessen Farbe c_1 ist) und t_2 (dessen Farbe c_2 ist). Daraus folgt, dass die Farbe von t $c_2 + c_1$ (Bezeichnung: $c_{(1,2)}$)ist. Dabei ist $c_{(1,2)}$ eine neue URI in $\boldsymbol{I}$. Auf die genaue Form dieser URI gehen die Autoren nicht ein. Folgende Ideen liegen den definierten Eigenschaften der „+" Operation zugrunde:

a) Ein implizites Tripel, das aus Tripeln der gleichen Farbe abgeleitet wurde, erbt die Farbe der implizierenden Tripel (Idempotenz).

b) Die Farbe eines impliziten Tripels wird eindeutig bestimmt durch die Farben der Tripel, aus denen es sich ableitet, und nicht durch die Reihenfolge in der die Inferenzregeln angewendet werden (Kommutativität und Assoziativität).

4.1.3 Verwaltung unsicherer Metadaten

Unsichere Daten werden meist durch die Angabe des Grads der Unsicherheit gekennzeichnet, in den meisten Fällen handelt es sich dabei um eine Wahrscheinlichkeit. Generell ist die Repräsentation unsicherer Metadaten in RDF über Quintupel denkbar. Hierbei wird einem Tripel zusätzlich zu seinem Named Graph noch eine Wahrscheinlichkeit angeheftet. Das Management probabilistischer Daten ist ein aktuelles Forschungsthema. Eine Lösungsvorschlag hierfür sind Uncertainty-Lineage Databases.

Uncertainty-Lineage Databases

In [ben06] werden Uncertainty-Lineage Databases (ULDBs) vorgestellt. Für das ULDB Datenmodell werden Unsicherheiten durch besondere Tupel dargestellt. Diese können für ein oder mehrere Attribute alternative Werte besitzen. Jede dieser Alternativen kann mit einem Vertrauenswert assoziiert werden. Auch die Existenz eines Tupels kann mit einer Unsicherheit behaftet sein, die ebenfalls über ein Vertrauensmaß spezifiziert werden kann. Aufbauend auf den genannten Attribut- und Vertrauenswerten stellt jede ULDB mehrere *possible-instances* dar. Diese werden manchmal auch *possible-worlds* genannt. Eine einzelne possible-instance ist eine reguläre relationale Datenbank. Um Daten mit Informationen über Ableitung in Verbindung zu bringen, wir das Konzept der Lineage (Abstammung), manchmal auch Provenance (Herkunft) genannt, eingeführt. Diese existiert in den zwei Ausprägungen *intern* und *extern*. Um interne Abstammung handelt es sich, wenn die Daten sich innerhalb der ULDB befinden. Von externer Abstammung spricht man, wenn es sich um Daten ausserhalb der ULDB handelt, oder um Daten von anderen datenproduzierende Entitäten wie Programmen oder Devices.

Zusammengefasst lässt sich somit sagen, dass ULDBs das Standard SQL relationale Modell durch folgende Merkmale erweitern: Durch die Angabe alternativer Werte lassen sich Unsicherheiten über Tupelinhalte spezifizieren. Durch Maybe-Annotationen können Unsicherheiten über die Existenz eines Tupels angegeben werden. Alternativen können mit numerischen Vertrauenswerten versehen werden. Durch Lineage können Tupel-Alternativen mit anderen Tupel-Alternativen in Verbindung gebracht werden.

4.2 Architekturentscheidungen / Gesichtspunkte für die Verwaltung

Dieser Abschnitt befasst sich mit den Möglichkeiten zur Verwaltung der Metadaten. Hierzu müssen betreffend der Speicherung und des Speicherorts der Daten unterschiedliche Gesichtspunkte betrachtet werden. Um mögliche Alternativen zur Verwaltung der Daten in ein sinnvolles Bild zu setzen, bildet die im generischen Arbeitspaket G1 erarbeitete allgemeine Referenzarchitektur die Grundlage. In D.G1.4A [WSM+09] wurde die Aletheia-Architektur spezifiziert. Abbildung 7 zeigt die resultierenden Kernservices und deren Interaktion.

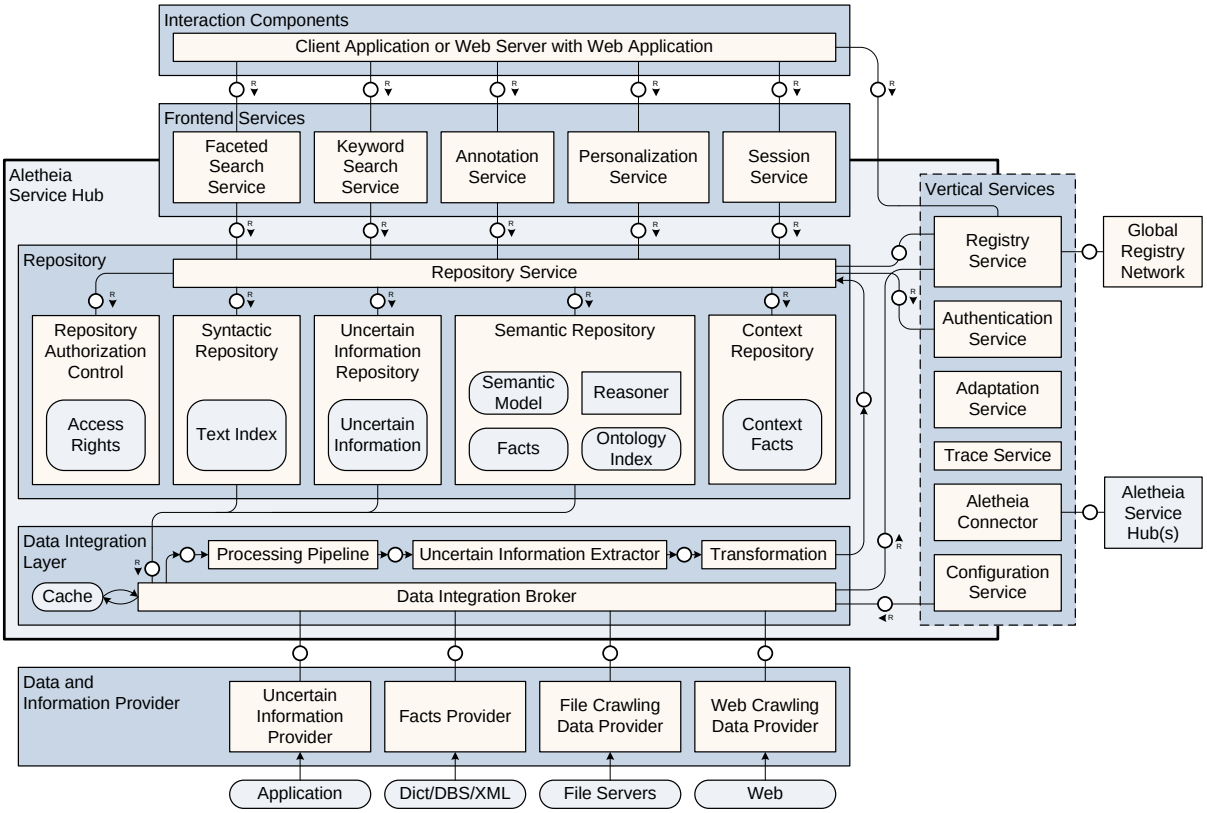

Abbildung 7: Kernservices und deren Interaktion aus [WSM+09]

In D.G1.4A werden die Funktionsblöcke des Repositories wie folgt beschrieben: Der Repository Service stellt die Funktionalität des Repository für die anderen Komponenten bereit. Über den Repository Service können Fakten und unsichere Informationen zu einer bestimmten Entität ausgegeben werden sowie Verweise auf Dokumente geliefert werden, in denen bestimmte Entitäten wahrscheinlich genannt werden oder in denen eine bestimmte Zeichenkette enthalten ist. Dafür nutzt dieser Dienst die darunterliegenden internen Komponenten.

Zu diesen internen Komponenten zählt das Syntactic Repository, das einen einfachen Dokumentenindex verwaltet und damit eine Schlagwort-basierte Suche ermöglicht. Daneben erlaubt das Uncertain Information Repository, Informationen zu verwalten, die nicht als zu 100% sicher gelten können, insbesondere automatisiert extrahierte Informationen. Das Semantic Repository schließlich verwaltet sowohl tatsächliche Fakten als auch das semantische Modell. Bei all diesen Repository-Komponenten findet ein semiföderierter Zugriff statt, d.h. externe Data Provider mit eigenem Index bzw. eigener Faktenbasis können

zur Anfragezeit angesprochen und deren Ergebnisse mit den intern gespeicherten Informationen des jeweiligen Repository integriert werden. Eine genauere Definition dieser Repository-Komponenten ist in Deliverable D.G4.2 [BWR+09] beschrieben.

Der Zugriff auf die Inhalte der Repository-Komponenten ist abhängig von der Nutzerrolle des jeweils Anfragenden. Zur Bestimmung der Zugriffsrechte für die Nutzerrollen wird eine Komponente benötigt, die die entsprechenden Berechtigungen verwaltet (Repository Authorization Control). Die Ermittlung der Benutzerrolle findet außerhalb des Repository (Authentication Service) statt.

Schließlich findet auch die Verwaltung von Kontext- und Sitzungsinformationen über ein Repository statt, dessen Informationen durch das Context Repository gekapselt werden. Dies ist eine logische Aufteilung, die konkrete Implementierung des Context Repository kann ggf. als Teil des Semantic Repository erfolgen.

Konkret geht es nun darum zu untersuchen, wie und wo die Metadaten in Aletheia abgelegt werden sollen. Hierfür werden zwei unterschiedliche Varianten betrachtet. Zum Einen die Verwaltung der Metadaten im semantischen Repository, zum anderen die Verwaltung der Metadaten über einen Index.

4.2.1 Verwaltung der Metadaten im semantischen Repository

Werden die Metadaten (gemeinsam mit den Produktdaten) im semantischen Repository gehalten, muss beachtet werden, dass es sich hierbei um sichere Metadaten handeln muss. Zur Verdeutlichung werden die Abläufe zur Speicherung und zum Zugriff anhand der Gesamtarchitektur erläutert.

Gewinnung und Abspeicherung der Metadaten

Die Metadaten werden, wie in Kapitel 3.4 beschrieben, durch den Data and Information Provider, den Data Integration Broker und den Uncertain Information Provider bereitgestellt. Um eine adäquate Speicherung des Metadatum zu gewährleisten muss im Anschluss überprüft werden, ob es mit Unsicherheiten behaftet ist. Ist dies nicht der Fall wird es im RDF-Store abgespeichert. Grundsätzlich geschieht dies über die Einfüge-/Speichermechanismen des RDF-Stores.

In welchem Format die Metadaten im RDF-Store gespeichert werden, hängt davon ab, welche Variante aus Abschnitt 4.1.1 bzw. 4.1.2 gewählt wird. Allgemein gilt hier: Die (sicheren) Metadaten kommen als Tripel aus der Extraktion. Diese können im Repository

- direkt als Tripel gespeichert werden, oder
- als Named Graphs (bzw. Quadrupel), oder
- als reifizierte Tripel.

Die Speicherung als reifizierte Tripel stellt nicht die Methode der Wahl da. Durch die Reifikation werden die ursprünglichen Nutzerdaten verändert. Dies hat zur Folge, dass diese dann nicht mehr normal angefragt werden können. Berücksichtigt man dies, werden die

Metadaten im RDF-Store somit im gleichen Format wie die Benutzerdaten abgelegt. Die Speichermechanismen für die Metadaten entsprechen denen der Produktdaten. Nach dem Speichern der Metadaten sind Benutzerdaten und Metadaten im semantischen Repository abgelegt und über bestimmte Mechanismen miteinander verknüpft. Der Zugriff auf die Metadaten kann grundsätzlich in gleicher Form erfolgen wie der Zugriff auf die Produktdaten. Abhängig von der Verknüpfung der Benutzerdaten und Metadaten können zusätzliche Mechanismen wie in Abschnitt 4.1.2 beschrieben notwendig sein.

Zugriff auf die Metadaten

Der Zugriff auf die Metadaten wird im Folgenden anhand einer beispielhaften Nutzeranfrage erläutert, die an den Repository Service geschickt wird. Dieser leitet die Anfrage an die verschiedenen unterliegenden Repositories weiter.

Anfrage: Gib mir alle Dokumente, von denen Person X der Autor ist.

1. Der Repository Service fragt das semantische Repository an, ob Aussagen existieren mit `PersonX istAutor Dokument`.

2. Im semantischen Repository wird nach entsprechenden Aussagen gesucht. Zudem wird nach abgeleiteten Fakten gesucht. Wenn also zum Beispiel Dokument eine Subklasse von Schriftstück ist, dann wird auch gesucht ob es Aussagen gibt mit PersonX istAutor Schriftstück. Die Ergebnisse werden an den Repository Service zurückgegeben.

4.2.2 Verwaltung unsicherer Metadaten

Wie bereits in Kapitel 4.1.3 erwähnt wurde, handelt es sich bei der Verwaltung unsicherer (Meta-)Daten um ein aktuelles Forschungsthema. Die konkrete Implementierung des Uncertain Repositories in Aletheia wird zum gegenwärtigen Zeitpunkt untersucht. Daher kann an dieser Stelle nicht genau beschrieben werden, wie die unsicheren Metadaten verwaltet werden und der Zugriff oder Anfragen aussehen. Denkbar ist die Verwaltung der unsicheren Daten in einem RDF-Store getrennt von den sicheren Daten im semantischen Repository. Hierbei ist zu beachten, dass aufgrund der Unsicherheiten die Schlussfolgerungsmechanismen des RDF-Stores nicht verwendet werden dürfen. Einen weiteren Lösungsansatz stellen probabilistische Datenbankmanagementsysteme dar, die im nächsten Abschnitt beschrieben werden.

4.3 Beschreibung von Tools und Methoden zur Verwaltung der in Aletheia anfallenden Metadaten

In diesem Abschnitt werden Tools betrachtet, mit denen die zuvor vorgestellten sicheren und unsicheren Metadaten verwaltet werden können.

4.3.1 Tools zur Verwaltung sicherer Metadaten

Die Verwaltung sicherer Metadaten kann in RDF-Stores erfolgen. Diese wurden in D.G4.2 bereits auf ihre für Aletheia allgemein wichtigen Merkmale hin untersucht. An dieser Stelle werden die zur Verwendung empfohlenen RDF-Stores aus [BWR+09], die zur Verwaltung von Metadaten geeignet sind, vorgestellt, sowie die jeweiligen Vorzüge oder Nachteile für die vorliegende Problemstellung herausgearbeitet.

Sesame

Sesame[16] ist ein Open Source Framework für RDF und RDFS, welches das Speichern, Abfragen (Querying) und Schließen (Reasoning) von RDF-Daten unterstützt. Sesame weist in vielerlei Hinsicht Ähnlichkeiten zu Jena auf, doch wohingegen bei Jena (gesehen als komplettes semantisches Framework) der Schwerpunkt auf der Unterstützung von Reasoning liegt, ist Sesame hingegen für die Speicherung und das Abfragen von RDF-Daten optimiert. Sesame bietet keine gesonderte Metadatenverwaltung an, unterstützt jedoch die Spezifikation und Verwendung von Kontext durch Quadrupel[17], so dass auf diese Weise auch Metadaten verwaltet werden können.

OntoBroker RDF

OntoBroker RDF bietet grundlegende Unterstützung für Quadrupel. Die Verwendung von Quadrupeln geht mit verschiedenen Konsequenzen einher. Ein Kontext-Identifikator muss eine URI oder eine IRI sein. Da keine Ontologieobjekte, und somit auch keine Standardaxiome erzeugt werden, sind SPARQL-Anfragen nur ausführbar, wenn das Inferenzing deaktiviert wird. OntoBroker RDF verwendet die Sesame-Parser und unterstützt daher sowohl die TriX als auch die TriG Serialisierung. Sollen Quadrupel bearbeitet werden kann dies über eine API-Erweiterung geschehen. Diese unterstützt das manuelle Hinzufügen und Löschen von Quadrupeln, sowie auch das manuelle Löschen mit einfachen Filterausdrücken (Platzhaltern). Die Darstellung von Beziehungen zwischen Fakten und Metadaten in Ontobroker wird über die Angabe unterschiedlicher Module gelöst, wie dies auch für MetaK erläutert wurde. Die Angabe unterschiedlicher Module für alle Fakten resultiert dementsprechend in einer sehr großen Menge an Modulen.

AllegroGraph

AllegroGraph ist ein Datenbank- und Applikations-Framework für die Erstellung von Semantic-Web-Applikationen. Es ist in der Lage, Daten und Metadaten als Tripel zu speichern, diese Tripel mit verschiedenen Anfrage-APIs abzufragen (z.B. SPARQL und Prolog) und RDFS++-Reasoning mit dem integrierten Reasoner durchzuführen. Allegrograph ermöglicht die Repräsentation von Sichten, Vertrauensfaktoren und Herkunft. Dies geschieht durch die Speicherung von Quintupeln, die aus fünf Slots bestehen. Die ersten drei Slots sind für Subjekt, Prädikat und Objekt des Tripels vorgesehen. Hinzu kommt ein Slot für den Named Graph und eine Unique ID, die durch Allegrograph zugeteilt wird. Der ID Slot dient primär internen administrativen Zwecken, kann jedoch auch von anderen Tripeln direkt referenziert werden. Genauer heisst das, dass eine Tripel-ID das Subjekt oder Objekt eines anderen Tripels sein kann. Auf diese Weise können auch unsichere Metadaten verwaltet werden. Ein Vorteil dieses Ansatzes ist, dass die Gesamtmenge der Tripel reduziert wird. Damit geht einher, dass sich die Ausführungs- beziehungsweise Antwortzeit einer Query verringert, da die einzelne Query mehr Daten abfragen kann. Der Nachteil des Ansatzes ist jedoch, dass die ursprünglichen Daten verändert werden und nicht mehr wie

[16] http://www.openrdf.org
[17] http://www.openrdf.org/doc/sesame2/users/ch08.html#d0e1218

gewohnt angefragt werden können. Das W3C Proposal sieht eine Verwendung des Named-Graph Slots für die Ballung der Tripel vor.

Mulgara

Mulgara ist ein System zur Verwaltung von Metadaten über Entitäten verschiedener Dokumente und entstand ursprünglich aus dem Kowari-Projekt. Es speichert solche Daten und kreiert Relationen zwischen ihnen. Die Informationen werden als Tripel, also Subjekt-Prädikat-Objekt-Aussagen verwaltet und orientieren sich am RDF-Standard. Metadaten können direkt als RDF importiert werden. Mulgara ist optimiert für die Speicherung und Abfrage von Metadaten. In der aktuellen Version werden Multi-Prozessoren unterstützt. Das Repository bietet volle Transaktionsunterstützung und permanente Integrität. In Mulgara zu verarbeitende Metadaten lassen sich als RDF/XML oder RDF/N3 im- und exportieren. Die Regelmengen können bei Bedarf angepasst werden. Da zum Zeitpunkt der Erstellung des Deliverables die Webseite von Mulgara nicht verfügbar war, konnten keine weiteren Informationen zur Repräsentation von Metadaten gewonnen werden.

Virtuoso

Virtuoso ist eine Datenintegrations- und Management-Lösung, die einheitlichen Zugriff auf SQL, RDF, XML, Web Services und Geschäftsprozesse ermöglicht. Durch seine Datenreplikations- und Synchronisationsmechanismen ermöglicht Virtuoso automatisierte Verteilung und Aktualisierung von SQL und Webinhalten über verteilte Virtuoso Server.

Virtuosos RDF-Unterstützung beinhaltet Unterstützung für die SPARQL-Anfragesprache. Zusätzlich sind eine Anzahl von Erweiterungen wie Pfadtraversierung und Business Intelligence Features enthalten. Weiterhin wurden Sicherheitsmechanismen basierend auf Virtuosos Unterstützung für Row-Level-Policy-Based Sicherheit, Custom Authentication, und Named Graphs integriert. Mithilfe eines Volltextindex werden RDF-Statement Objektwerte indiziert.

4.3.2 Tools zur Verwaltung unsicherer Metadaten

Wie in Kapitel 4.1.3 beschrieben, erfordert die Verwaltung unsicherer Metadaten die Möglichkeit, Wahrscheinlichkeiten mit den Daten mitzuführen. Dieser Abschnitt befasst sich daher mit verschiedenen Probabilistischen Datenbankmanagementsystemen. Der Fokus all dieser Systeme liegt weniger auf der Repräsentation von Aussagen, bzw. Tripeln, sondern auf dem Anheften von Unsicherheiten an Daten.

MayBMS[18]

MayBMS ist ein DBMS für unsichere, beziehungsweise probabilistische Daten, das durch eine Modifikation des Postgres Server Backends realisiert wurde. Dies bringt verschiedene Vorteile mit sich. So kompiliert MayBMS auf den gleichen Plattformen wie Postgres. Zudem können Postgres APIs und Middleware verwendet werden. Hierzu zählen etwa ODBC, JDBC oder auch PHP. SQL wird vollständig unterstützt. Dabei entspricht die Performanz entspricht

[18] http://www.cs.cornell.edu/bigreddata/maybms/

der von Postgres auf vollständigen Daten. MayBMS unterstützt Updates, Transaktionen und Recovery in vollem Umfang und ist Open Source verfügbar[19].

ORION DBMS[20]

Orion DBMS ist ein DBMW für unsichere Daten das integrierte Unterstützung für probabilistische Daten bietet. Orion unterstützt sowohl Attribut- als auch Tupel-Unsicherheiten mit beliebigen Wechselbeziehungen und unterscheidet sich so von anderen Datenbanken für unsichere Daten. Dieses Feature ermöglicht es diskrete, aber auch kontinuierliche PDFs (Probability Density Function) in angemessener Weise zu handhaben. Das zugrundeliegende Modell ist geschlossen unter den grundlegenden relationalen Operatoren und konsistent mit der Possible World Semantik. Wie auch andere probabilistische DBMS baut Orion auf Postgres auf. Orion bietet effiziente Zugriffsmethoden um unsichere Daten anzufragen. Hierzu gehören auch drei verschiedene Indexstrukturen, die auf R-Trees, Signature Trees und invertierten Indexen basieren.

Zusätzliche Statistiken über probabilistische Datentypen können erfasst und ausgenutzt werden und auf diese Weise das Optimieren von Queries, Join Algorithmen und Selektivitäts-Schätzungen ermöglichen. Für die grafische Visualisierung von und statistische Inferenz über unsichere Daten bietet Orion eine Integration mit PL/R.

Trio[21]

Trio ist ein von der Universität Stanford [MTdK+07] [Wid08] entwickeltes DBMS für die Haltung von Daten, samt ihrer Unsicherheiten und Herkunft. Trio basiert auf dem erweiterten relationalen Modell ULDB (Uncertainty-Lineage Database), und unterstützt eine SQL-basierte Query Sprache (TriQL).

[19] http://maybms.sourceforge.net/
[20] http://orion.cs.purdue.edu/
[21] http://infolab.stanford.edu/trio/

5 Methoden zur Nutzung von Qualitätsmetadaten

Das Ziel der Erhebung und Verwaltung von Qualitätsmetadaten ist, es den Nutzern und Anwendungen zu ermöglichen, die Relevanz der dargestellten Produktinformationen besser bewerten zu können, als es ohne den Einsatz von Qualitätsmetadaten möglich wäre. In Kapitel 2.4.2 wurde bereits bei jedem Metadatum auf mögliche Nutzungen eingegangen. In diesem Kapitel werden diese Möglichkeiten noch einmal unter den Gesichtspunkten der Nutzung durch das Aletheia-System selbst (5.1) sowie durch den Endanwender (5.2) betrachtet.

5.1 Nutzung durch das Aletheia-System selbst

Die Nutzung der Qualitätsmetadaten durch das System wird entlang der Verarbeitungskette beschrieben. Informationen werden aus Ressourcen extrahiert, semantisch annotiert, im Repository abgelegt und schließlich bei einer Suche in einer nach Relevanz sortierten Liste zurückgegeben.

Anhand der bei der Extraktion gewonnenen *Confidence* einer Information wird im Repository Service entschieden, ob es sich um einen Fakt oder eine unsichere Information handelt. Dementsprechend wird sie im Semantic oder Uncertain Repository abgelegt wird.

Das Crawling-Datum (*DateCrawling*), die letzte Änderung (*DateChanged*) und die Dynamik einer Ressource (*SourceDynamics*) bestimmen die Aktualität der extrahierten Informationen. Um die im Repository verwalteten Produktinformationen möglichst aktuell zu halten, können die drei Qualitätsmetadaten von den Data oder Information Providern verwendet werden, um entsprechende Monitoring-Strategien mit verschiedenen Frequenzen einzusetzen, welche die Ressourcen auf Änderungen zu überprüfen. Für den erneuten Zugriff auf die Ressource und die Datenquelle wird die *URISource* benötigt.

Aus der Angabe eines Autors und dem Kontext der Information lässt sich mit einem Reputationssystem die Reputation des Autors berechnen. Dies ist zwar kein Bestandteil von Aletheia, wäre aber eine interessante Erweiterung der Funktionalität. Die Reputation einer Person ist ihr Ruf, hat jemand eine hohe Reputation in einem Fachgebiet, so hat er einen guten Ruf, was sich auf die zu erwartende Qualität seiner Dokumente und den daraus extrahierten Produktinformationen positiv auswirkt.

Das Ranking der Informationen, d.h. eine Bewertung und Sortierung ihrer Relevanz bezüglich einer konkreten Anfrage eines Nutzers ist eine weitere wichtige Aufgabe des Systems. Neben dem in diesem Schritt berechneten *RelevanceScore* kann ein Großteil der erfassten Metadaten mit in die Berechnung einbezogen werden. Hat eine Information A eine niedrigere *Confidence* als Information B, so ist sie voraussichtlich weniger relevant für die Anfrage. Laut Cohen et al. [CDZ08] ist das Datum das vorherrschende Sortierkriterium von Nutzern. Hinter dem Datum verbirgt sich die Aktualität der Information, welche durch die Aggregation von *SourceDynamics*, *DateCrawling* und *DateChanged* ermittelt werden kann. Liegen z.B. *DateChanged* und *DateCrawling* dicht bei einander, jedoch weit in der Vergangenheit bezüglich des aktuellen Datums und ist die Quelle hochgradig dynamisch, so kann davon ausgegangen werden, dass die extrahierte Produktinformation nicht mehr aktuell ist. Liegt *DateChanged* hingegen weit in der Vergangenheit und *DateCrawling* sowie das Datum der Anfrage des Nutzers nahe bei einander, so kann die Schlussfolgerung

getroffen werden, dass die Information im Repository höchstwahrscheinlich aktuell ist und dass sie sich auch seit ihrer letzten Änderung (*DateChanged*) nicht mehr geändert hat. Schlussfolgerungen dieser Art sind, anders als beim Reasoning über Fakten, mit einem einzukalkulierenden Restrisiko eines Fehlers behaftet. Die Qualitätsmetadaten *DateValidFrom* und *DateValidUntil* können ebenfalls in die Berechnung der Relevanz eingehen. Sucht ein Nutzer z.B. nach seinen aktuellen Vertragsbedingungen eines Wartungsvertrages, so werden die AGBs abgelaufener Verträge nicht mit als Ergebnis zurück gegeben.

5.2 Nutzung durch den Endanwender (Mensch)

Zunächst können alle Qualitätsmetadaten vom Anwender direkt in die Anfrage einbezogen werden, um die Ergebnismenge entsprechend einzuschränken. So können z.B. nur Informationen aus einer bestimmten Datenquelle wie dem ServIS bei ABB angefordert werden, wofür intern die *URISource* verwendet wird. Suchen sind ebenso nach Dokumenten bestimmter Autoren oder aus bestimmten Zeiträumen möglich. Eine vom Verfasser abhängige Suche nach Ressourcen ist unter anderem in AP.A4 von Interesse: wenn man Autoren von techn. Handbüchern, Service Berichten etc. ermitteln könnte, würde man auch andere von Ihnen erstellte Dokumente als 'höherwertig' einstufen können.

Eine weitere Möglichkeit der Nutzung durch den Anwender besteht in der Präsentation der Metadaten in den Suchergebnissen, um zusätzliche Informationen bereitzustellen.

Wird dem Endanwender im Suchergebnis neben den ermittelten Informationen jeweils noch die Ressource angegeben, aus der die Information stammt, so kann er bei Interesse das Quelldokument (Ressource) als Ganzes betrachten oder Informationen über die Datenquelle (z.B. File-Server) erlangen, auf der die Ressource gespeichert wird. Hierfür ist die Angabe der *URISource* nötig.

Im Allgemeinen vertrauen Menschen ihnen bekannten Personen mehr, als fremden Leuten, von denen sie noch nie etwas gehört haben. Bekommt eine Nutzerin Alice vier Dokumente präsentiert, die ihrer Anfrage entsprechen und sind diese mit den jeweiligen Autoren (*Author*) annotiert, so kann Alice auf ihren Erfahrungsschatz zurückgreifen und die Dokumente ihr bekannter Autoren mit gutem Ruf zuerst lesen. Nicht zuletzt wird der Autor einer Information in AP.A3 ausgewertet, um ihn auf einen Vertrauenswert abzubilden. Es macht einen großen Unterschied, ob die Position und der Zustand eines Produktes von einem vertrauenswürdigen Sensor ermittelt wurden, oder ob „irgend jemand“ der Urheber ist.

SourceDynamics kann als Zusatzinfo für den Nutzer verwendet werden, wenn dieser die Quelle der präsentierten Informationen nicht kennt und gern mehr in Erfahrung bringen möchte, ohne die Quelle selbst in Augenschein zu nehmen. Die zusätzliche Angabe der letzten Änderung (*DateChanged*) hilft ihm darüber hinaus, die Aktualität einer Information zu betrachten. Aus den Änderungsdaten verschiedener Dokumente, welche die gleiche Information enthalten kann ein Nutzer unter Umständen den Urheber einer Information ausfindig machen. Diese Funktionalität wird u.a. im Arbeitspaket AP.A1 benötigt.

Der *RelevanceScore* wird zur Sortierung der Ergebnisse einer Anfrage verwendet und kann dem Nutzer mit angezeigt werden, damit er besser einschätzen kann, ob diese Information noch relevant für ihn ist.

Die *Confidence* einer Information, d.h. die Selbsteinschätzung der Extraktionsqualität einer Information, kann dem Nutzer ebenfalls präsentiert werden. Hierbei ist jedoch besonders die Interpretation des Qualitätsmetadatums durch den Anwender zu beachten. Es muss geprüft werden, in wie weit die Angabe den Nutzer in seiner Suche unterstützt oder ob sie ihn eher verwirrt, da voraussichtlich ein Großteil der Informationen unsicher ist, was im Anwender ein negatives Gefühl hervorrufen könnte.

Für alle Qualitätsmetadaten wird daher im weiteren Projektverlauf geprüft, wie sie den Anwendern zugänglich gemacht werden und ob manche nur erfahrenen Benutzern bereitgestellt werden. Sind diese Fragen geklärt, wird die in T.G4.3 zu entwickelnde Anfragesprache entsprechend modifiziert.

6 Zusammenfassung

In diesem Deliverable werden die in Aletheia benötigten Qualitätsmetadaten eingeführt und beschrieben. Qualitätsmetadaten ermöglichen es einem Nutzer sowie dem System selbst, die Qualität, Frische und Herkunft der verwalteten Produktinformationen zu bewerten. Das Ziel der Erhebung und Verwaltung von Qualitätsmetadaten ist, es den Nutzern und Anwendungen zu ermöglichen, die Relevanz der dargestellten Produktinformationen besser bewerten zu können, als es ohne den Einsatz von Qualitätsmetadaten möglich wäre.

Basierend auf den Anforderungen der fünf Anwendungspartner erfolgt in Kapitel 2 zunächst eine Identifikation der zu betrachtenden Qualitätsmetadaten und ihrer Funktionalitäten. Eine ausführliche Definition verwendeter Begriffe trägt zur Verwendung einer einheitlichen Terminologie innerhalb des Projektes bei. Der Schwerpunkt des Kapitels liegt auf einer ausführlichen Analyse der Typen von Metadaten, die in Aletheia verwendet werden.

Kapitel 3 beschreibt allgemeine und konkrete Methoden zur Gewinnung von Qualitätsmetadaten und beleuchtet die Erweiterungen der Datenintegrationsplattform. Dabei wird ausgehend von der Architektur der Datenintegrationsplattform ein allgemeiner und wiederverwendbarer Prozess zur Gewinnung von Qualitätsmetadaten definiert. Es wird aufgezeigt, wie dieser Prozess in die bestehenden Abläufe der Informationsgewinnung in Arbeitspaket AP.G3 integriert werden kann und welche Komponenten hierfür zuständig sind.

Um die gewonnen Metadaten in Aletheia zu verwalten, folgen in Kapitel 4 Grundlagen und eine State-of-the-Art-Betrachtung zur Verwaltung von Qualitätsmetadaten, die in Aletheia einheitlich mit den übrigen Metadaten verwaltet werden sollen. Sowohl die Gewinnung, die Speicherung als auch der Zugriff auf die Daten zählen zu Aspekten der Verwaltung. Die Art und Weise der Datenspeicherung wirkt sich signifikant darauf aus, zu welchem Zeitpunkt Metadaten generiert werden müssen und wie performant auf sie zugegriffen werden kann. Aus diesem Grund wird auf die Gesichtspunkte, die sich mit der Speicherung befassen, in diesem Kapitel das Hauptaugenmerk gelegt.

Eine Betrachtung der Methoden zur Nutzung von Qualitätsmetadaten in Kapitel 4 rundet das vorliegende Dokument ab. Als Nutzer der Metadaten werden sowohl die Nutzer des Aletheia-Systems, sowie das System selbst betrachtet.

7 Literaturverzeichnis

[ben06] *ULDBs: databases with uncertainty and lineage*. VLDB Endowment, 2006.

[BWR+09] A. Becker, M. Walther, S. Reichert, J. Hladik, and F.Dau. D .g4.2 repository-spezifikation. Technical report, ontoprise GmbH, 2009.

[BYRN99] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1st edition, May 1999.

[CDZ08] Sara Cohen, Carmel Domshlak, and Naama Zwerdling. On ranking techniques for desktop search. *ACM Trans. Inf. Syst.*, 26(2):1–24, 2008.

[DHB+09] F. Dau, J. Hladik, A. Becker, S. Brockmans, R. Korf, M. Erdmann, and M. Niemann. Aletheia d.g4.1 modellierungsmethodik und globales semantisches modell. Technical report, SAP, 2009.

[DSSS09] R. Dividino, Sergej Sizov, Steffen Staab, and B. Schüler. Querying for provenance, trust, uncertainty and other meta knowledge in rdf. *Journal on Web Semantics*, 2009. Special issue on "The Web of Data".

[FFP+09] Giorgos Flouris, Irini Fundulaki, Panagiotis Pediaditis, Yannis Theoharis, and Vassilis Christophides. Coloring rdf triples to capture provenance. In *8th International Semantic Web Conference (ISWC2009)*, October 2009.

[MTdK+07] Michi Mutsuzaki, Martin Theobald, Ander de Keijzer, Jennifer Widom, Parag Agrawal, Omar Benjelloun, Anish Das Sarma, Raghotham Murthy, and Tomoe Sugihara. Trio-one: Layering uncertainty and lineage on a conventional dbms. In *Third Biennial Conference on Innovative Data Systems Research (CIDR 2007)*, pages 269–274, Asilomar, USA, 2007.

[Nau02] Felix Naumann. *Quality-driven query answering for integrated information systems*. Springer-Verlag New York, Inc., New York, NY, USA, 2002.

[RWU09] Robert Rieger, Maximilian Walther, and David Urbansky. D.G3.2 Spezifikation der Grundfunktionalitäten. Technical report, SAP AG, 2009.

[SWR09] Marcus Schramm, Maximilian Walther, and Robert Rieger. D.G3.1 State-of-the-Art und Anforderungsanalyse. Technical report, SAP AG, 2009.

[Tol06] K. Tolle. *Semantisches Web und Kontext – Speicherung von und Anfragen auf RDF-Daten unter Berücksichtigung des Kontextes*. PhD thesis, Johann Wolfgang Goethe – Universität Frankfurt am Main, 2006.

[Wid08] Jennifer Widom. Trio: A system for data, uncertainty, and lineage. In *Managing and Mining Uncertain Data*. Springer, 2008.

[WS96] Richard Y. Wang and Diane M. Strong. Beyond accuracy: what data quality means to data consumers. *J. Manage. Inf. Syst.*, 12(4):5–33, 1996.

[WSM+09] M. Wauer, D. Schuster, T. Münch, D. Urbansky, and M. Walther. D.g1.4a spezifikation der aletheia-architektur zur föderierten verwaltung unterschiedlicher produktinformationen. Technical report, Technische Universität Dresden, 2009.