



Das Graphenmodell von RDF

Wir sind es gewohnt, MetaDaten mit dem <Meta> Tag von HTML im Head von HTML Dokumenten unterzubringen.

Diese Art der Übermittlung bringt eine Reihe von Beschränkungen mit sich. Einige Probleme sehen wir im folgenden kleinen Beispiel:

```
<HTML>
<HEAD>
<META NAME="DC.subject.msc" CONTENT="(Scheme=msc91) 19-XX">
<META NAME="DC.Format" CONTENT="application/x-dvi">
<META NAME="DC.Identifier" CONTENT="(SCHEME=url)
    http://www.math.uiuc.edu/ K-theory/0316/vv.dvi">
<META NAME="DC.Format" CONTENT="application/postscript">
<META NAME="DC.Identifier" CONTENT="(SCHEME=url)
    http://www.math.uiuc.edu/K-theory/0316/vv.ps.gz">
<META NAME="DC.Identifier.mirror" CONTENT="(SCHEME=url)
    http://www.mathematik.uni-osnabrueck.de/K-theory/0316/vv.ps.gz">
</HEAD>
```

Fehlende Gruppierungsmöglichkeit: Welcher Identifier gehört zu welcher Formatangabe.
Fehlende Typisierungsmöglichkeit: 19-XX ist ein Klassifikationscode. Dies ist durch die Schemeangabe nicht klarzumachen. Es könnte auch die Klartextfassung: Algebraic K-Theory folgen.

Vermischung von Objekttypisierung und Beziehungsverfeinerung: Die Beziehung "identifier" wird nicht durch den Umstand enger gefasst, dass das Präfix "http://www.mathematik.uni-osnabrueck.de/K-theory" den Standort eines Mirrors wiedergibt.

HTML Meta Probleme:

- Assoziativ
- Kommutativ
- Folge 1:1 Problem, Personen/ Korporationen Problem, Beschreibungsschwierigkeiten bei Multi-Part und abstrakten Objekten.
- Warwick Frame Work: Koexistenz, aber keine Kooperation von Schemata realisierbar
- Kein Recycling von Semantik
- Beziehungen zwischen Tag - Namen nicht beschreibbar
- Verfeinerungen: PunktPunkt=Punkt Problem
- Schemata: Link Tag nicht funktional an Meta Tag gebunden.
- Namenskollisionen müssen durch zusätzliche Konventionen vermieden werden.
- Versionsproblem
- Vokabular Inflation

Fehlende Strukturierungsmöglichkeiten wohin man blickt.

RDF Modell und Syntax Spezifikation

Entstehung

Das Resource Description Framework RDF ist ein Ergebnis der Arbeiten des W3 Consortiums an der Web - Infrastruktur für MetaDaten. Das grundlegende Papier ist die am 22.2.1999 vom W3 herausgegebene Recommendation über die Modell und Syntax Spezifikation [RDF MSS] .

Die RDF MSS wurde von einer Arbeitsgruppe unter der Leitung von Eric Miller (OCLC, DC Directorate) und Bob Schloss (IBM) erarbeitet. Herausgeber der Endversion sind Ora Lassila (Nokia) und der W3 Mitarbeiter Ralph Swick.

Motivation und Zielsetzung

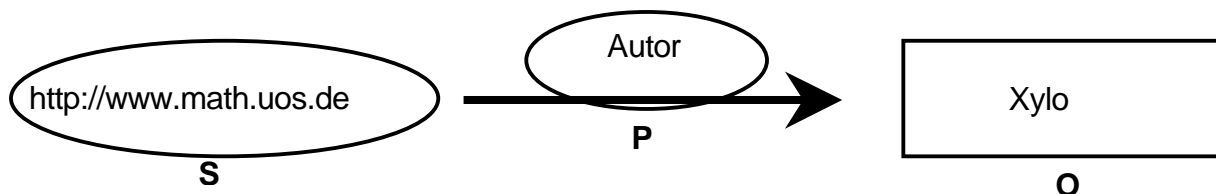
Die Notwendigkeit zu einer solchen Infrastruktur haben wir eingangs an den Defiziten von HTML Meta erläutert.

- Grundbedingung für Interoperabilität mit den übrigen Aktivitäten zur Stärkung der Web Infrastruktur ist eine Bindung an XML, die die Rolle des physikalischen Übermittlungsmediums einnimmt.
- Gleichzeitig muss die Verwaltung der kodierten Semantik durch Datenbanken im Grundsatz gesichert sein.
- Last but not least: Die Spezifikation muss die Entwicklung graphisch orientierter Design- und Ausgabewerkzeuge ermöglichen.

Im Detail sind diese Ansprüche nicht deckungsgleich. Es resultieren einerseits Einschränkungen andererseits Relationen, die bei einer Beschränkung des Blickwinkels auf einen der drei Aspekte hätten vermieden werden können.

Ausgangspunkt

Ausgangspunkt der Überlegungen zu RDF sind die Vergleichssätze menschlicher Sprachen: O ist der P von S.



Bei diesem Ausgangspunkt liegt wie in einer Nusschale eine Datenbankverwaltung durch Tripel und eine graphische Darstellung durch gerichtete Graphen mit Bezeichnungen (Labeled Directed Graphs) nahe.

Aufgabe der Spezifikation:

„Konstruiere eine Klasse A von wohlgeformten XML Dokumenten, eine Klasse B von Tripelmengen und eine Klasse C von Labeled Directed Graphs zusammen mit Transformationsregeln zwischen ihnen.“

Vorgehensweise der Spezifikation

RDF MSS erreicht dieses Ziel auf folgendem Weg:

- XML Klasse A: Definiert in BNF in RDF MSS 6. Formal Grammar [6.1] - [6.34]
- Tripel Klasse B: Definiert in RDF MSS 5. Formal Model for RDF item 4 und 6. Formal Grammar nach [6.34].
- Labeled Directed Graphs Klasse C: Implizit durch Transformationen in RDF MSS 2.- 4. definiert.

Theoretisches

Definition: RDF Objekt: Ein 3- Tupel (a, b, c), wobei a aus A, b aus B und c aus C ist, heißt ein RDF Objekt, wenn aus einer der drei Komponenten die beiden anderen durch die Transformationsregeln hervorgehen.

Definition: RDF Dokument:

Zwei RDF Objekte sind Repräsentanten desselben RDF Dokuments, wenn sie in wenigstens einer Komponente (Repräsentation) gleich sind.

Ziel des Vortrags: Die Graphenrepräsentation von RDF Dokumenten

Die konkrete Beschreibung der XML Dokumente, die zu einem RDF Objekt gehören, ist Inhalt anderer Vorträge.

- Der Wunsch nach Existenz einer XML Darstellung stellt die stärkste Einschränkung bei der Konstruktion von RDF Objekten dar, führen aber auch zu Äquivalenzsetzungen.
- Labeled Directed Graphs führen zu den stärksten Äquivalenzsetzungen. Auch der Wunsch nach Beschreibung durch Tripel hat Äquivalenzen zur Folge. Diese können als weitere Einschränkung für die Graphen beschrieben werden.
- Umformulierung und Konsequenz:
- RDF Dokumente können durch Isomorphieklassen von Labeled Directed Graphs beschrieben werden. Jede solche Isomorphieklasse bestimmt das RDF Dokument eindeutig.

Notation für Namen

Der Wunsch nach Interoperabilität mit allgemeiner Web Infrastruktur hat - teilweise bereits durch die Bindung an XML - auch Konsequenzen bezüglich der Notation.

Will man einem Gegenstand einen Web-Bezeichner geben, so muss dazu der IETF RFC 2396: Uniform Resource Identifier (URI): Generic Syntax [RFC2396] verwendet werden.

RDF "versteht" also grundsätzlich folgende Notation nicht: 'Es bezeichne "this:that" die Menge der reellen Zahlen.'

Im Grundsatz würde: 'Es bezeichne "./this:that" die Menge der reellen Zahlen.' "verstanden" werden. Das Problem liegt in der besonderen Verwendung des : der URI Syntax.

Was ist eine Ressource?

Benannte Ressourcen

- RDF beschränkt sich nicht auf die Beschreibung von Objekten, die es irgendwo im Web gibt. RDF beschränkt lediglich den Zeichenvorrat für Objektnamen auf (URI) und unterwirft den Zeichenvorrat den in RFC2396 gemachten syntaktischen Bedingungen.
- Die benannten Ressourcen eines RDF Objektes kann man aufteilen in die referenzierten und die im Objekt neu benannten Ressourcen. Das Verhalten dieser beiden Typen ist in der Graphenrepräsentation gleich. Sie spielt daher in diesem Vortrag - im Gegensatz zur XML Repräsentation keine weitere Rolle.

Die Verwendung von Bezeichnern hat eine sehr sinnvolle praktische Konsequenz: Es ist nicht möglich in einem RDF Objekt einen Bezeichner für verschiedene Dinge zu verwenden.

Unbenannte Ressourcen

In menschlicher Sprache ist es möglich über Dinge zu reden, ohne ihnen einen Namen zu geben.

- A besitzt ein Ding. Dieses Ding ist ein Buch.
- ...Gestern Nacht kam ein Männlein in ihre Kammer und hat alles Flachs für sie gesponnen....

(Wir wissen alle zu welchem Drama sich die Namensfindung ausgeweitet hat.)

Wollen wir also zulassen, dass Aussagen über Dinge mit unbekanntem Namen oder auch Dinge, die wir nicht benennen wollen oder Dinge, die (derzeit) keinen Namen besitzen, gemacht werden, so müssen wir Mechanismen einbauen, die eine Identifizierung und Unterscheidung aus dem Kontext zulassen.

In der RDF MSS wird ein extremer - aber sicherer - Standpunkt eingenommen:

- Namenlose (privat gehaltene) Objekte sind zulässig. Sofern nicht jenseits aller Zweifel die Identität feststeht, werden namenlose Objekte nicht einander gleichgesetzt.

Literals

- In der RDF MSS werden Aufmacher die Literals des RDF Objektes genannt

Auch bei der Definition von Literals macht sich die Bindung an XML bemerkbar:

- Die Literals eines RDF Objektes bilden eine Menge. Die Elemente dieser Menge sind selbst beliebiges wohlgeformtes XML. Im weiteren bezeichnen wir mit Literals stets endliche Mengen von wohlgeformtem XML.

Theoretisches

Wir sind nun soweit, die Tripel im Sinne des RDF MSS formal beschreiben zu können. Dazu führen wir zunächst den Begriff eines RDF Modells ein.

Definition: RDF Modell: Gegeben sei eine Menge R . Eine Teilmenge P von R , eine Menge L und eine Teilmenge S von $R \times P \times (R \cup L)$. Das Vier-tupel $M = (R, P, L, S)$ heißt ein RDF Modell.

Die Menge R wird die Menge der Ressourcen von M , P die Menge der Properties (Eigenschaften), L die Menge der Literals in M und S die Menge der Statements (facts) von M genannt.

Definition: Feines RDF Modell: Es sei $M = (R, P, L, S)$ ein RDF Modell. Für R sei eine Zerlegung in disjunkte Teilmengen X, Y gegeben und eine Darstellung von X als Teilmenge von (URI - absolut).

Das Sechstupel $N = (R, P, L, S, X, Y)$ heißt ein feines Modell. Die Statements (a, b, c) mit b in X heißen die Tripel des feinen Modells.

Definition: Minimales feines Modell: Ein feines Modell N heißt minimal, wenn die Vereinigung des Bildes von erster Projektion, zweiter Projektion und dritter Projektion der Tripel von N die Ressourcen R umfasst, die zweite Projektion den Durchschnitt von P mit X und die dritte Projektion L umfasst.

Bemerkung: Zu jedem feinen RDF Modell gibt es ein minimales feines Modell.

Minimale Modelle haben die Eigenschaft, dass jede Ressource in einem Statement vorkommt und jedes Literal wenigstens einmal adressiert wird.

Definition: Es seien N_1, N_2 zwei feine Modelle. N_1, N_2 heißen modellisomorph, wenn $X_1 = X_2$ und es eine Bijektion $f: Y_1 \rightarrow Y_2$ gibt.

Definition: Modelläquivalenz: Zwei feine Modelle N_1 und N_2 heißen modelläquivalent, wenn die zugeordneten minimalen feinen Modelle modellisomorph sind.

Satz: Durch RDF MSS 6 wird für jedes RDF XML Dokument ein bis auf Modelläquivalenz eindeutiges feines RDF Modell konstruiert.

Definition: Das Tripelmodell von RDF besteht aus den minimalen feinen Modellen in deren Äquivalenzklasse ein durch RDF MSS 6 konstruiertes Modell liegt.

Bemerkung: Um genau zu sein müsste man in dieser Definition sagen: ...nach RDF 1.0 konstruiertes Modell liegt.

RDF Graphen

Die formale Beschreibung von Tripelrepräsentationen von RDF Dokumenten hört sich grauslich kompliziert an. Etwas informeller kann man aber zumindest im Falle endlicher Ressourcenmenge formulieren:

- Eine Tripeldarstellung erhält man durch Auflistung aller im XML RDF Objekt gemachten Statements. Dabei erhalten die privaten Ressourcen flüchtige Identifier. Die Namen benannter Ressourcen werden in ihrer vollständig aufgelösten Form angegeben. Literals sind als solche erkennbar.
- Der Übergang zur Graphendarstellung wird uns vom Zwang, flüchtige Identifier hinschreiben zu müssen, die dann aber doch als solche irrelevant sind, befreien.

Bemerkung: Wir beschäftigen uns im weiteren nur mit Tripel- und Graphendarstellungen von RDF Objekten, die in ihren XML Darstellungen die "aboutEachPrefix" Konstruktion nicht benutzen.

Mathematisch gesehen können die RDF Ressourcen aus diesen gutartigen XML Dokumenten in aufzählender Mengenschreibweise gegeben werden.

Ein Tripel haben dann die Gestalt:

- ([URI(Subject)], [URI(predicate)], [URI(Object)]) oder
- ([URI(Subject)], [URI(predicate)], Literal)

Es hat sich eingebürgert, die neu definierten Ressourcen mit

- `online#gegebenerName`

und die unbenannten Ressourcen mit

- `genidNummer`

zu kennzeichnen. Dabei ändert sich die `genidNummer` der unbenannten Ressource von Parserlauf zu Parserlauf, um anzudeuten, dass es sich um eine Bezeichnungskrücke handelt.

Vom Tripel zum einfachsten Graphen

Ein Graph besteht aus Knoten und Verbindungsstücken. Bei einem gerichteten Graphen hat das Verbindungsstück eine Richtung. Ein benannter Graph trägt Text- oder andere Merkmale an Knoten und Pfeilen.

- Für ein Tripel (URI1, URI2, URI3) setzen wir je eine Ellipse für URI1 und URI3 aufs Blatt Papier. In die Ellipsen tragen wir URI1 bzw. URI3 ein.
- Wir ziehen einen Pfeil von der URI1 Ellipse zur URI3 Ellipse. Auf den Pfeil schreiben wir URI2.
- Für ein Tripel (URI1, URI2, Literal) verfahren wir fast genauso. Statt der Ellipse für URI3 malen wir ein Rechteck.

Von Tripelmengen zu Graphen

Wir haben beschrieben wie wir mit der Abarbeitung einer Tripelmenge beginnen. Wir beschreiben jetzt wie wir mit weiteren Tripeln fortfahren.

- Sei (URIx, URly, URlz) das nächste Tripel. Falls für URlx und URlz noch keine Ellipse existiert, verfahren wir wie oben.
- Gibt es für URlx oder URlz schon eine Ellipse, so verbinden wir diese mit einem (weiteren) Pfeil auf den wir URly schreiben.
- Für Literals verfahren wir analog.

Da wir angenommen haben, dass die Tripelmenge endlich ist, sind wir nach endlich vielen Schritten fertig.

Bislang haben wir uns nur von der zufälligen Reihenfolge bei der Auflistung der Tripel befreit. Der entscheidende Schritt erfolgt jetzt:

- Nach getaner Arbeit entfernen wir alle Label der Form genidNummer.

Das Ergebnis ist jetzt von der erzwungenen Wahl von Namen für private Ressourcen unabhängig. Natürlich ist die Lage der Ellipsen und Rechtecke, der Größe und Länge und Krümmung der Verbindungsstücke immateriell.

Jedem steht es somit frei, für sich besonders schöne Graphenvisualisierungen zu wählen. Es ergibt sich ein reiches Betätigungsfeld für Werkzeugproduzenten.

Theoretisches

In völliger Strenge ist die Graphenrepräsentation eines RDF Objektes nicht die soeben beschriebene Visualisierung, sondern die am Schluss von Formalia II beschriebene Äquivalenzklasse minimaler feiner RDF Modelle. Damit ist man auch frei von der "aboutEachPrefix" Beschränkung. Allerdings lässt sich der Graph nicht mehr durch eine Menge in aufzählender Schreibweise notieren. Lassen wir ... bei der Visualisierung zu, so verallgemeinert die zuvor beschriebene Prozedur.

Sinn der Graphenrepräsentation/Visualisierung

Zu einem nützlichen Werkzeug und nicht nur zu theoretischen Überlegungen wird die Graphendarstellung, wenn wir Bedingungen angeben können unter denen ein Graph ein RDF Graph ist, d.h. wenn es ein XML Dokument gibt, dessen Graph der gegebene ist.

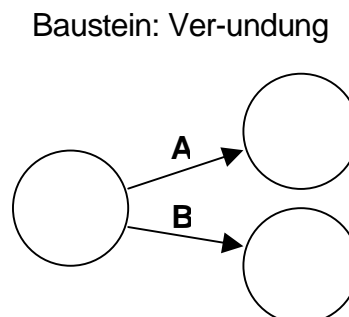
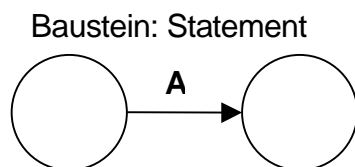
Solche Bedingungen wollen wir jetzt formulieren. Sie sind beim Design von MetaDaten Profilen außerordentlich nützlich.

Es macht einen erheblichen Unterschied, ob wir mit benannten Ressourcen oder privaten hantieren wollen. Bei den privaten Ressourcen muss sich deren Identität aus dem Kontext ergeben.

Vorgehensweise

- RDF Graphenbausteine
- Zusammensetzungsregeln für Bausteine

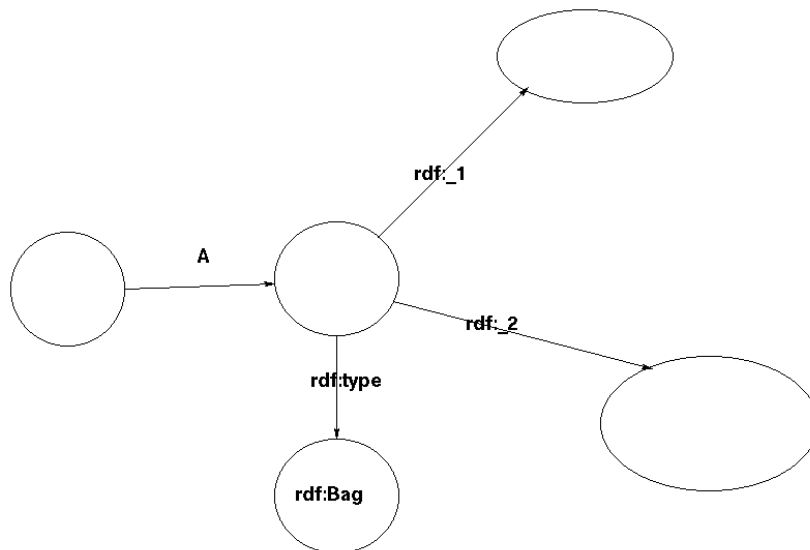
Insbesondere beim zweiten Punkt unterscheiden wir zwischen benannten und unbenannten Ressourcen.



Beide Aussagen können für sich alleine stehen. Volle Information wird nur bei Beachtung beider erhalten. Es kann eine beliebige (endliche) Menge von Statements mit

identischem Subjekt erzeugt werden. Unbenannte Objekte können nur von einer Property angesteuert werden. Eine Ausnahme ergibt sich bei der Reifizierung.

Baustein: Bag - Einkaufstüte (Sack)



Die Aussagen ergeben nur zusammengenommen einen Sinn. Ihre Reihenfolge ist irrelevant. Es kann eine beliebige (endliche) Anzahl von Aussagen gebündelt werden. Die von den Zählerelementen `rdf:_i` angesteuerten Ressourcen heißen die Objekte des Sacks. Der Ausgangsknoten eines Sacks kann aber muss nicht benannt sein.

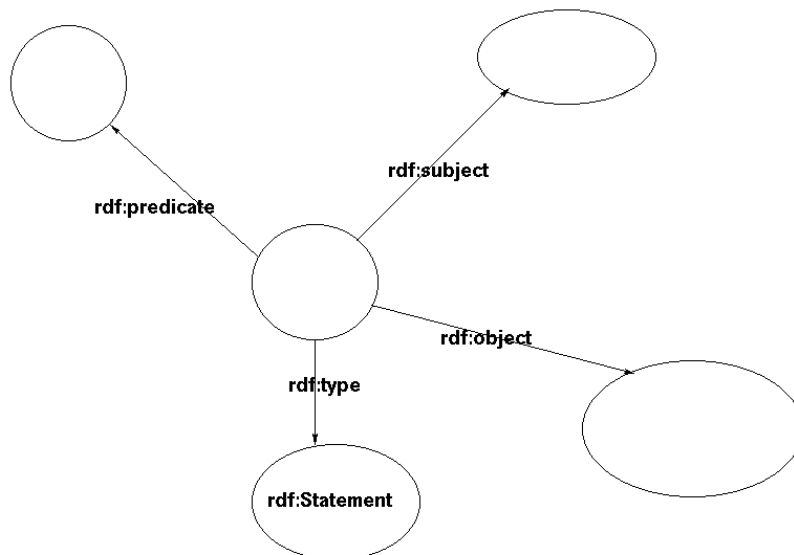
Baustein: Seq - Sequence

Die Graphendarstellung hat statt des Labels Bag das Label Seq. Die Reihenfolge ist bei Seq wichtig. Dieses Konstrukt eignet sich besonders für Listen. Nützlich ist es, die Art der Ordnung zu beschreiben.

Baustein: Alt - Alternative

Zur Erfüllung der eingehenden Eigenschaft genügt ein Zweig einer Alternative. Die Graphendarstellung unterscheidet sich nur im Label Alt vom Baggraphen. Alternativen müssen wenigstens ein Element enthalten. Bags und Sequenzen dürfen leer sein.

Baustein: Reifizierung



Der Baustein Reifizierung verursacht oft begriffliche Schwierigkeiten.

Die Standardanwendung der Reifizierungskonstruktion liegt bei Aussagen über Aussagen. Ihre Handhabung insbesondere im Zusammenhang mit unbenannten Ressourcen, kann schwierig sein.

Definition: Jede Ressource mit einer `online#Name` URI oder eine unbenannte Ressource, für die die angegebenen 4 Statements gesetzt sind, wird eine Reifizierung genannt.

Subjekt, Prädikat und Objekt Statement müssen dabei genau einmal gesetzt sein.

- Die Objekt Ressource von `rdf:predicate` ist per Definition die Property der Reifizierung.
- Die Semantik von Subject, Predicate, Object, Type sind von RDF fest definiert.
- `Rdf:type` muss immer eine Ressource als Ziel haben.
- Ist das Statement (Subjekt, Prädikat, Objekt) ebenfalls im RDF Objekt enthalten, so heißt die Reifizierung eine Reifizierung des gegebenen Statements.
- Handelt es sich bei Subjekt und Objekt um benannte Ressourcen, so kann beim Aufbau eines RDF Graphen zuerst eine Reifizierung gesetzt werden und erst anschließend das einfache Statement (S,P,O). Dasselbe gilt falls O ein Literal ist.
- Ist S oder O unbenannt, so muss - falls man (S,P,O) überhaupt in den Graphen einbringen will - zuerst (S,P,O) gesetzt werden.

Die weiteren Punkte zur Reifizierung betreffen Sonderfälle, die im Zusammenhang mit unbenannten Ressourcen Kopfzerbrechen bereiten können. Sie können in einem ersten Durchlauf überschlagen werden.

Ist O unbenannt, so kann (S,P,O) nur eine Reifizierung tragen. Ist P eine der Eigenschaften `rdf:subject`, `rdf:predicate`, `rdf:object` oder `rdf:type` und S vom Typ `Statement`, so kann es eintreten, dass nicht weiter reifiziert werden kann (siehe unten).

Ist S oder O unbenannt, so kann nur eine benannte Reifizierung unabhängig von weiteren Konstrukten hinzugefügt werden.

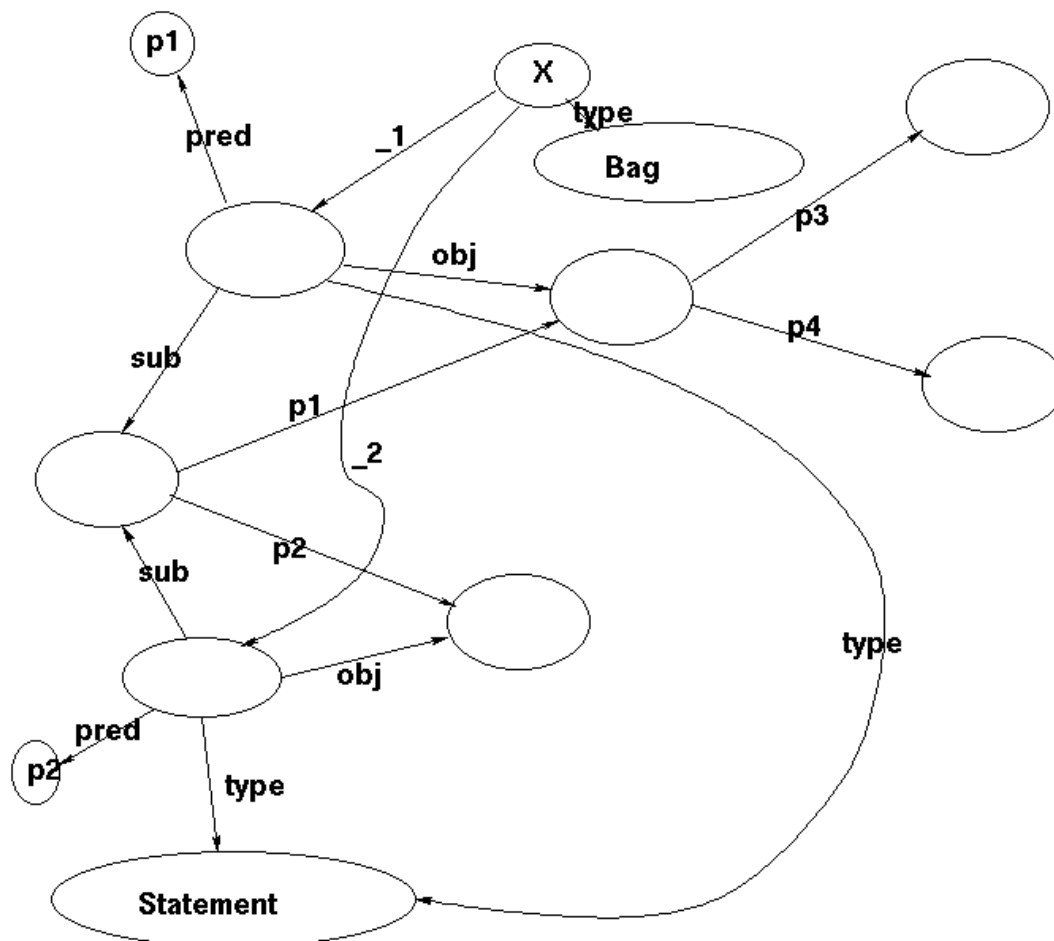
Die von einem Subjekt (unmittelbar) abgehenden Statements kann man gleichzeitig reifizieren. Gegebenenfalls bereits konstruierte Reifizierungen werden respektiert. Die neu hinzukommenden Reifizierungen sind unbenannt. Die Ressourcen der auf diese Weise beschriebenen Reifizierungen sind die Objekte einer mit einer `online#Name` benannten Ressource vom `rdf:type` eines Bag.

Die `rdf:subject`, `rdf:predicate` und `rdf:object` Bögen von unbenannten Reifizierungen aus dem letzten Schritt, können selbst nicht reifiziert werden.

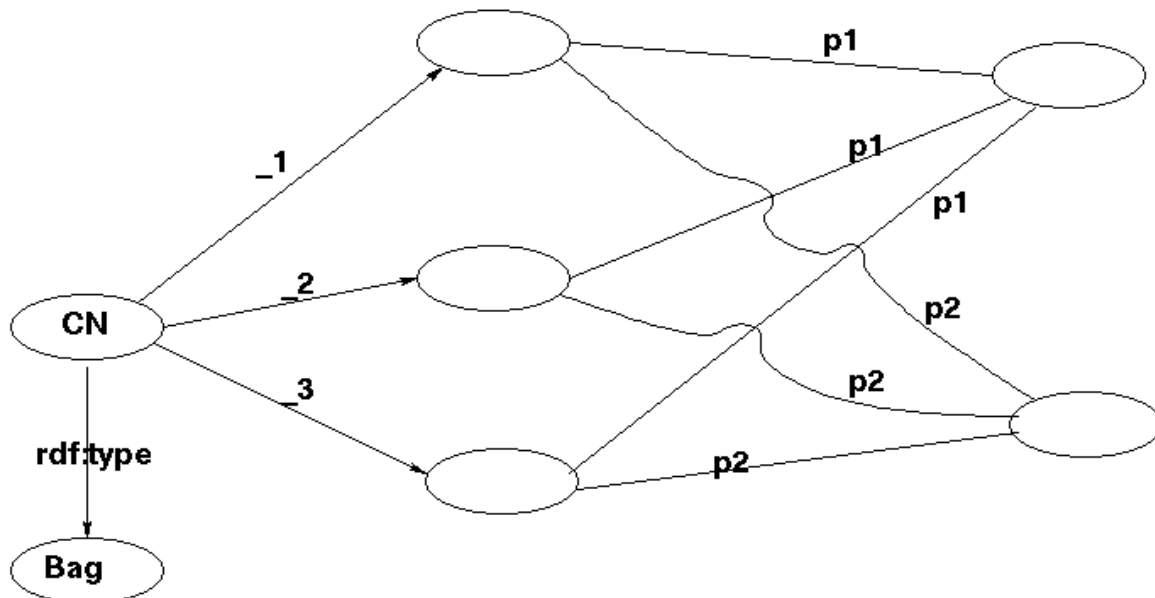
Als Ergebnis wollen wir zusammenfassen:

- Reifizierungen existieren in allen "vernünftigen" Fällen.
- Reifizierungen sind nicht immer eindeutig.
- In der Regel wird man nur bereits vorhandene Statements reifizieren wollen.

Baustein: Reifizierung im Sack



Baustein: Distributive Referenten



Als Containertypen sind auch Seq und Alt zulässig. Die Anker (Subjekte) beim distributiven Referenzieren sind die durch die Zählelemente ausgewiesenen Ressourcen. Literals können natürlich auch distributiv nicht referenziert werden.

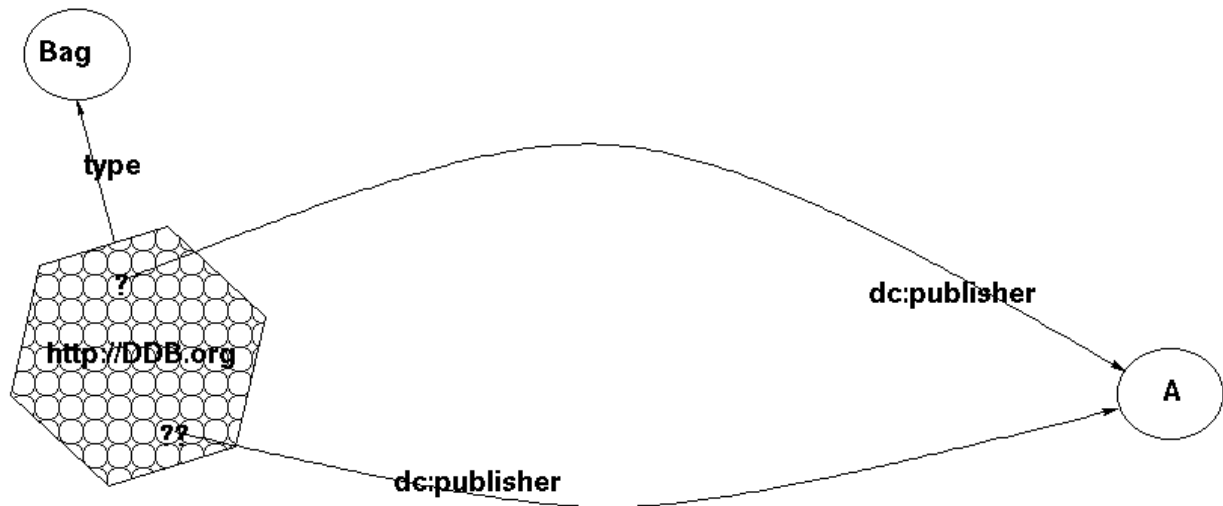
Baustein: Komponenten von unbenannten Ressourcen

Will man Aussagen zu unbenannten Ressourcen verketteten, so muss beachtet werden, dass abgesehen von der Reifizierung, eine unbenannte Ressource stets nur Ziel einer einzigen Eigenschaft sein kann.

Erweiterter Baustein: URI Präfix Distribution

Ein Präfix stellt man sich als Anfang von URI's vor. Präfixe verhalten sich dual zu Fragmentidentifiern. Es wird bei der Entwicklung etwa des Präfixes <http://www.mathematik.uni-osnabrueck.de> zur URL <http://www.mathematik.uni-osnabrueck.de/gibtEsNicht/> erwartet, dass es sich bei [.../gibtEsNicht/](http://www.mathematik.uni-osnabrueck.de/gibtEsNicht/) um die URL etwa eines files handelt. Ob - und wenn ja wie - ein Name mit "Etwas" verknüpft ist, ist nicht die Sorge von RDF. Die Meinung bei URL's vom Typ <http://> ist: Wenn das http Protokoll in der Lage ist eine Verbindung zu einem Objekt seines Gültigkeitsbereichs herzustellen, dann wird über dieses Objekt in RDF die dann folgende Aussage getroffen.

Ein Labeled Directed Graph der diesen Sachverhalt symbolisiert könnte etwa wie folgt aussehen:



Literatur:

RDF MSS: Resource Description Framework (RDF): Model und Syntax Specification
<http://www.w3.org/TR/1999/REC-rdf-syntax-19990222>

RFC 2396: Uniform Resource Identifiers (URI): Generic Syntax, August 1998
<http://www.hut.fi/u/jkorpela/rfc/2396/full.html>