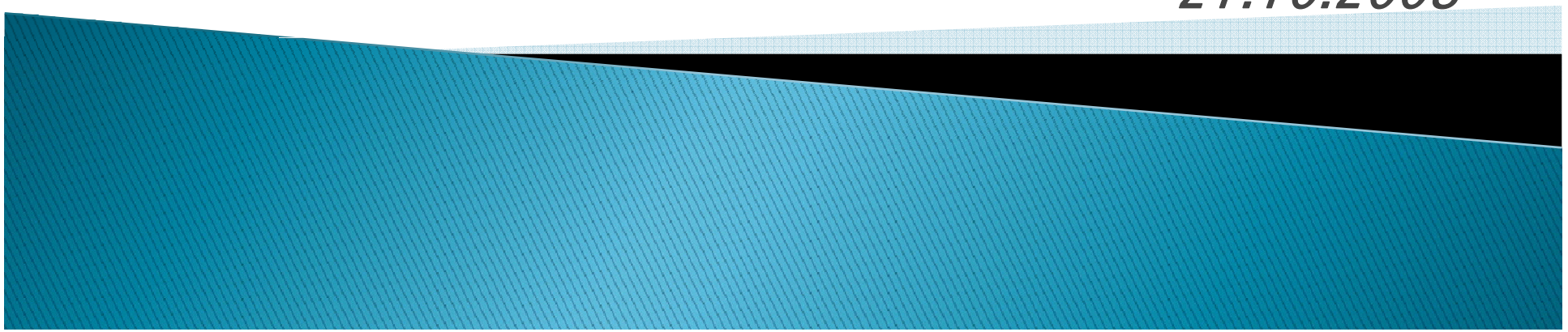
Team-Projekt

„Entwicklung einer RDF Suchmaschine“

Veranstalter: Lehrstuhl DBIS – Prof. Georg Lausen
Betreuer: Thomas Hornung, Michael Schmidt

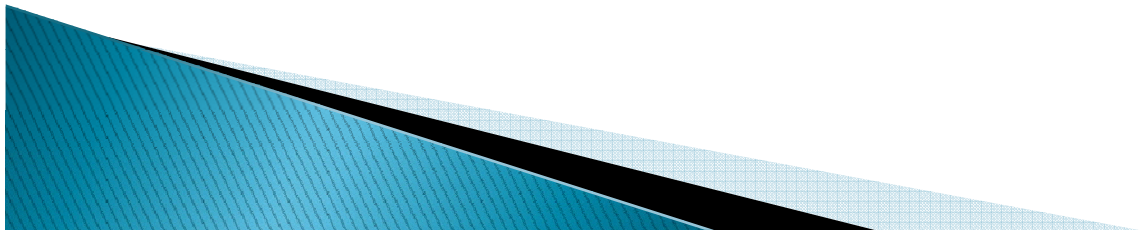
21.10.2008



Anforderungen

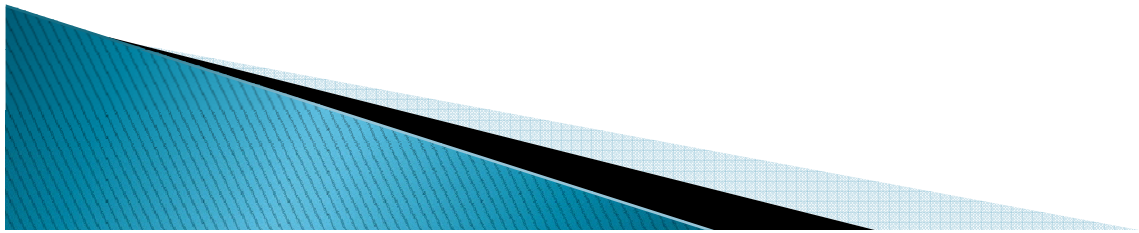
- Laut Studienordnung
  - Master/Diplom: 16ECTS/15KP
  - Entspricht: 480 Semesterstunden = **34h/Woche p.P.**
  - Teamgröße: 3-4 Studenten
  - Schriftliche Ausarbeitung: ca. 15-25 Seiten p.P.
  - Mündlicher Präsentation: ca. 15min p.P.

ANMELDUNG DER PROJEKTTEILNAHME
BEIM PRÜFUNGSAMT IST
ERFORDERLICH UND VERBINDLICH



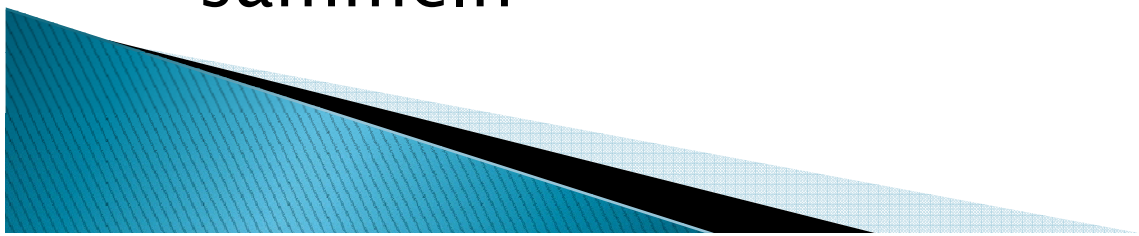
Organisation

- Zeit und Ort:
  - Dienstag 14–16 Uhr (c.t.)
  - Raum: SR 01–016, Geb. 101
  - **Anwesenheitspflicht für alle Teilnehmer**
- Zu Beginn: wöchentliche Treffen
- Später: 2-wöchentliche Treffen mit Kurzpräsentationen beider Teams; ggf. Besprechung von Problemen
- Weitere individuelle Termine auf Anfrage



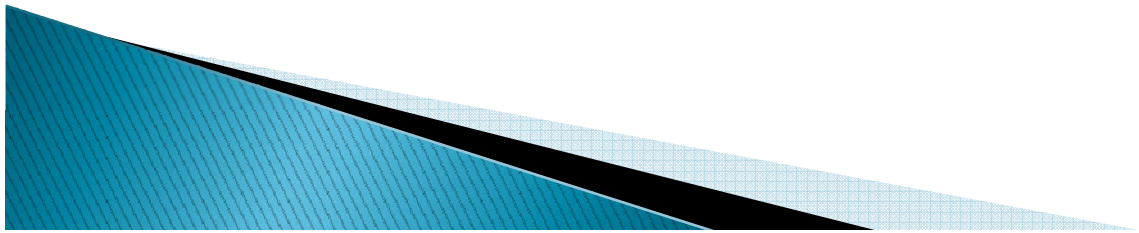
Ziele

- Gemeinsame Arbeit an einem großen Projekt
- Eigenständiges Recherchieren und Arbeiten
- Verbesserung der individuellen Programmierfähigkeiten (hier: Java, PHP, HTML)
- Einarbeiten in neue Themen (hier: RDF und Suchmaschinen)
- Probleme bei größeren Projekten Kennen und Lösen lernen
- Erfahrungen im Umgang mit Datenbanken sammeln



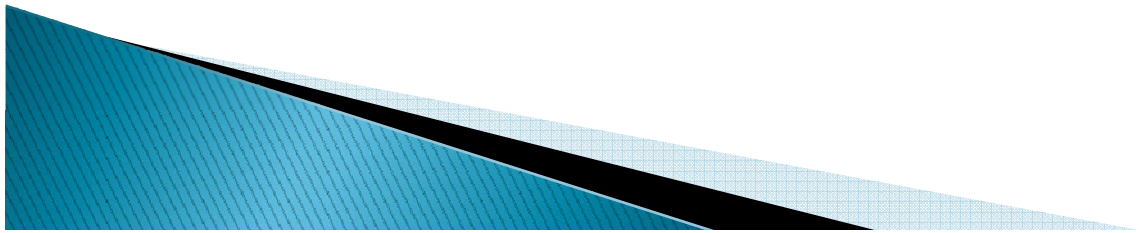
Schlüsselfaktoren zum Erfolg

- Gute Team-interne Organisation
  - Aufteilung von Verantwortung
  - Erfüllen von Verantwortung
  - Einhalten von Fristen
  - Gegenseitige Hilfe und Unterstützung
  - Saubere Definition von Schnittstellen
- Nutzen entsprechender Software (z.B. SVN)
- Einbringen individueller Fähigkeiten
- Spezialisierung auf Teilgebiete, ohne den Blick für das Ganze zu verlieren



Benotungsgrundlagen

- Insbesondere (aber nicht ausschließlich)
  - Umfang und Schwierigkeit der geleisteten Arbeit/Implementierung
  - Teamleistung: ein gelungenes Projekt wirkt sich in der Regel positiv auf die Noten einzelner Teammitglieder aus
  - Rolle und Mitarbeit im Team (Koordination etc.)
  - Qualität des Codes (Formatierung, Dokumentation)
  - Individuelle Ausarbeitung
  - Mündlicher Vortrag



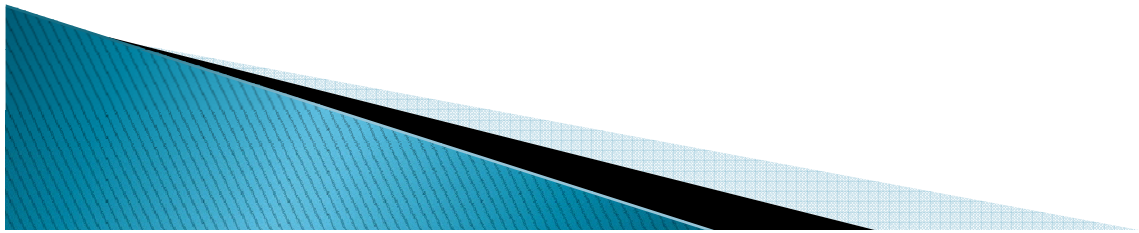
Aufgabenstellung

- „Entwicklung einer RDF-Suchmaschine“

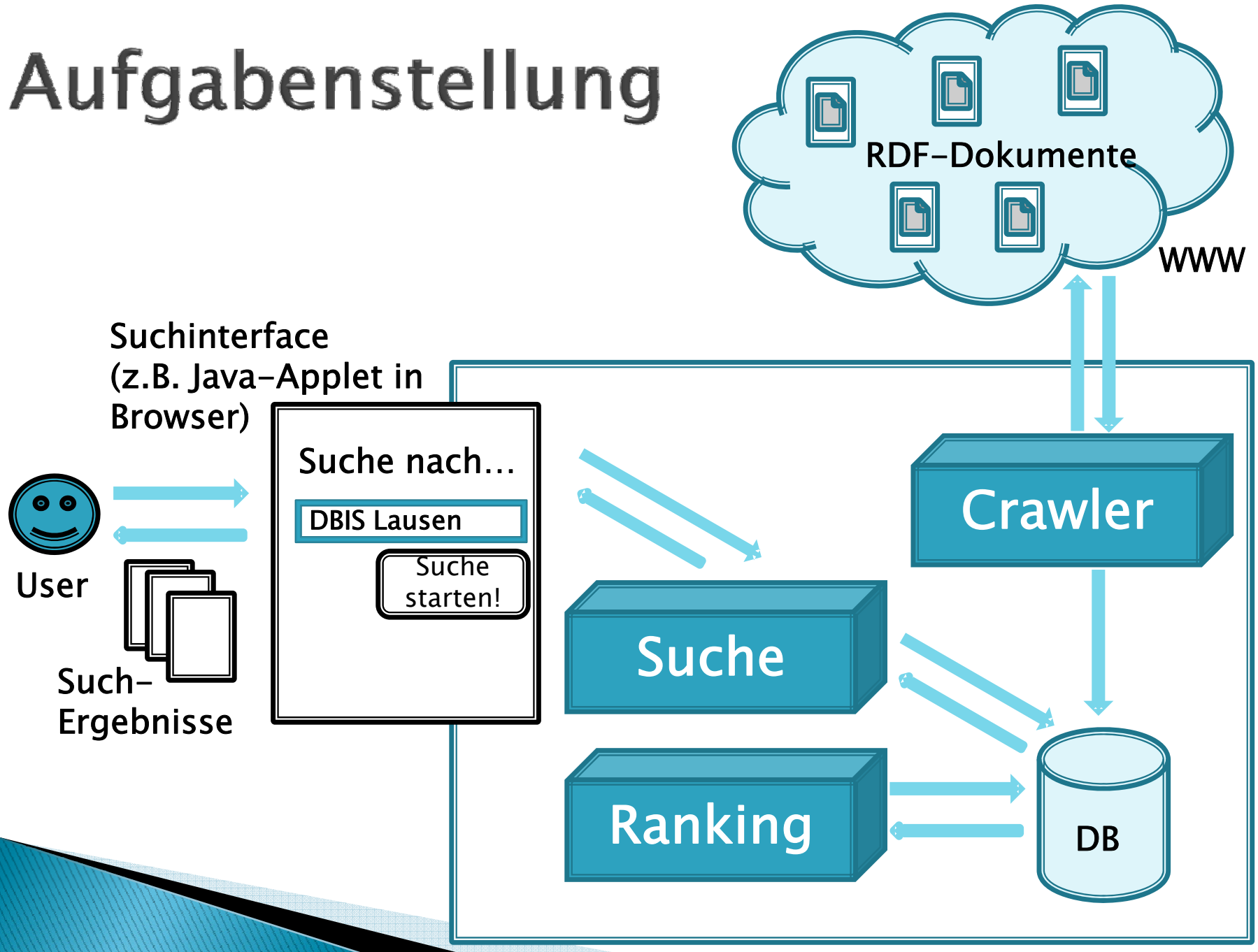
  Komponente 1:
  Crawler

  Komponente 2:
  Ranking-Schema für RDF

  Komponente 3:
  Design von Indexstrukturen und effiziente Suche auf den extrahierten und gerankten RDF-Daten

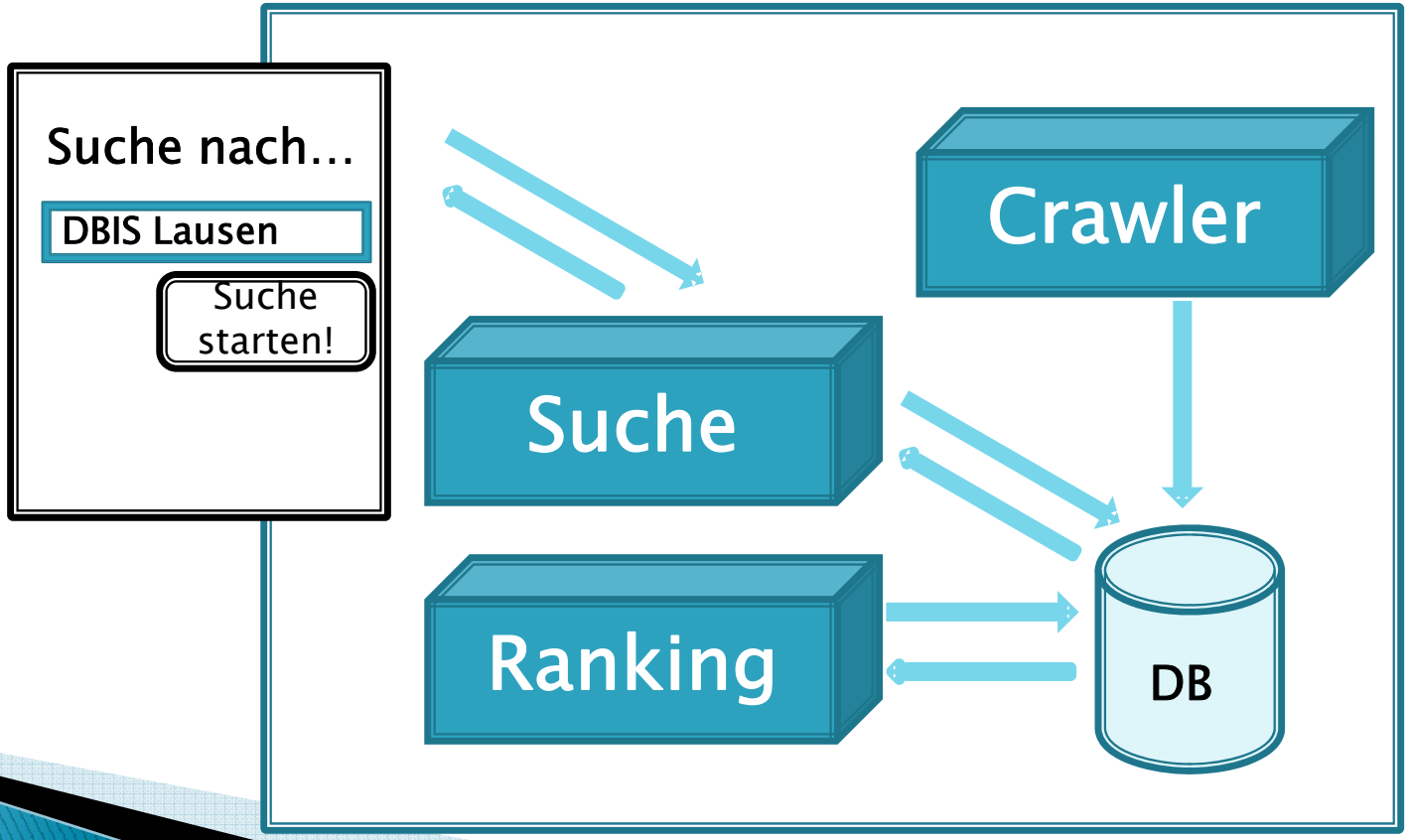


Aufgabenstellung



Aufgabenstellung

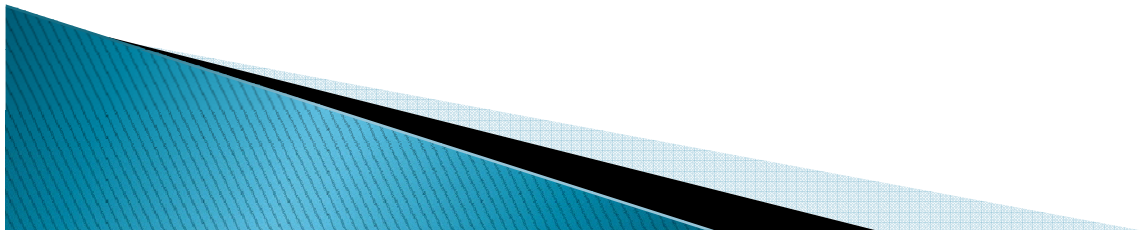
- Am Ende: Ein einziges voll funktionsfähiges System, das alle Komponenten beinhaltet



Komponente 1: RDF-Crawler

- Ziel
  - Extraktion von RDF-Daten aus dem WWW
  - Speichern dieser Daten in der Datenbank
  - Automatisches Verfolgen von RDF-Links, um weitere Dokumente zu finden
  - Intelligente Suchstrategien (z.B. Vermeidung von Duplikaten, parallele Anfragen etc.)

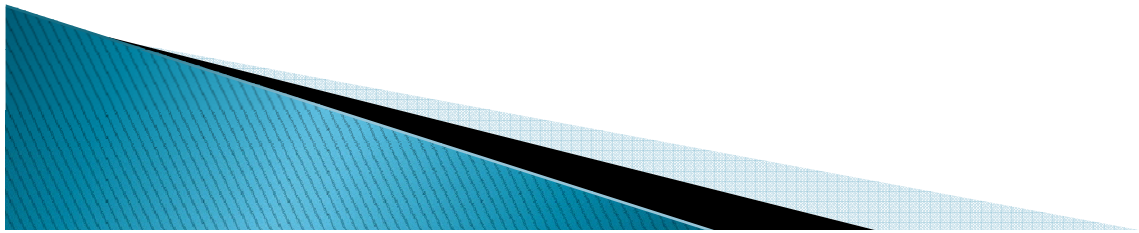
Mehr zum Thema Crawling und Indexstrukturen zur Suche am nächsten Dienstag!



Einführung RDF

- Datenformat
- Vom W3C standardisiert, ausführliche Informationen unter http://www.w3.org/RDF/
- Ursprüngliche Idee: Annotation von Webseiten mit maschinenlesbaren Meta-Informationen
- Datenformat basiert auf Tripeln der Form

(Subjekt, Prädikat, Objekt)

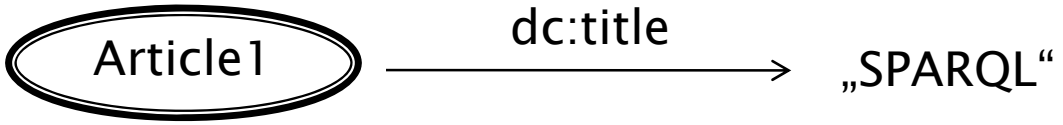


RDF Tripel

- Jedes Tripel repräsentiert einen Wissensfakt, z.B.

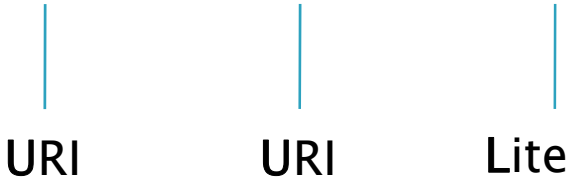
  (Article1, dc:title, „SPARQL“)

- Darstellung eines Tripels als eine gerichtete Kante in einem Graphen möglich



RDF Tripel

- Formale Definition:
  - Gegeben drei Mengen:
    U – Uniform Resource Identifiers (URIs)
    B – Blank Nodes („leere Knoten“)
    L – Literale
  - Ein RDF Tripel ist eine Element aus der Menge
    $(B \cup U) \times U \times (B \cup L \cup U)$
  - Beispiel:
    (Article1, dc:title, „SPARQL“)



RDF Datenbanken

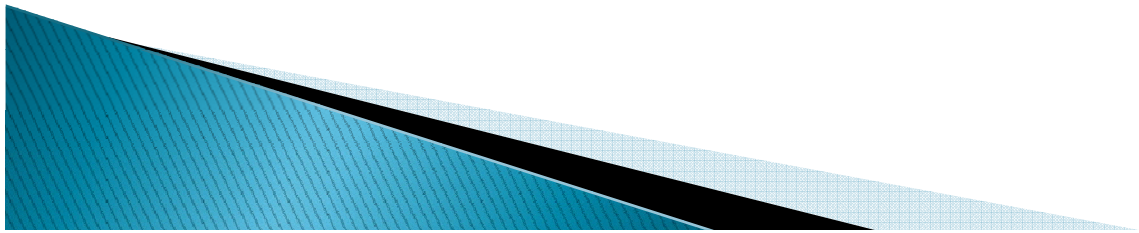
- Eine RDF Datenbank D (auch „Graph“ genannt) ist eine Menge von RDF Tripeln, formal

$$D \subseteq (B \cup U) \times U \times (B \cup L \cup U)$$

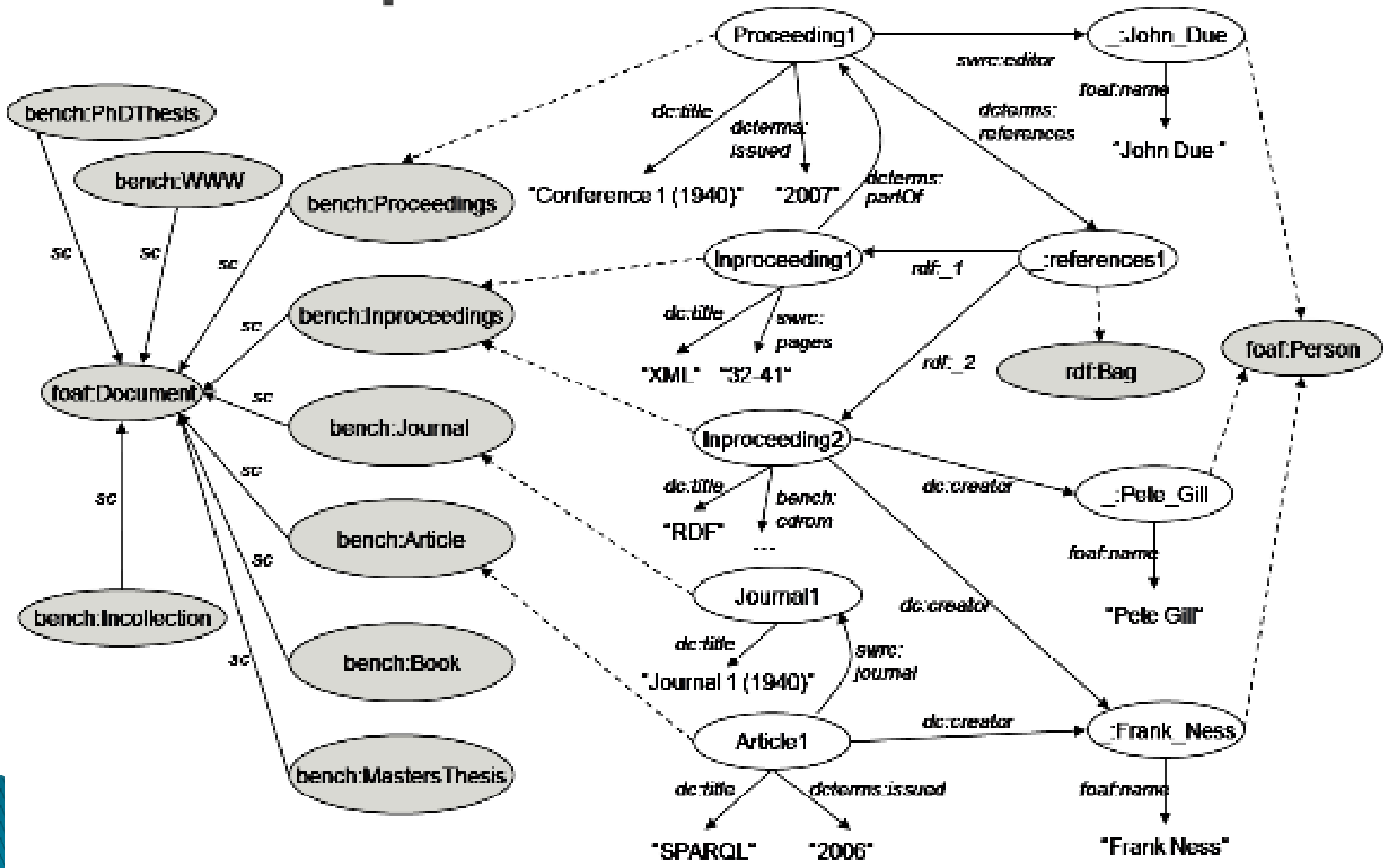
- Beispiel:

D = { (Article1, dc:title, „SPARQL“),
(Article1, dcterms:issued, „2000“),
(Article1, swrc:journal, Journal1),
… }

- Serialisierung von RDF Daten in verschiedenen Formaten möglich, z.B. NTriples, N3, RDF/XML, …



RDF Graph


Proceeding1
_:John_Due
swrc:editor
foaf:name
"John Due"
dcterms:
references
dc:title
dcterms:
issued
"Conference 1 (1940)"
"2007"
dcterms:
partOf
bench:PhDThesis
bench:WWW
bench:Proceedings
sc
sc
sc
Inproceeding1
rdf:_1
:references1
bench:Inproceedings
dc:title
swrc:
pages
"XML"
"32-41"
rdf:_2
rdf:Bag
foaf:Person
foaf:Document
sc
bench:Journal
Inproceeding2
dc:title
bench:
cdrom
"RDF"
...
dc:creator
_:Pele_Gill
foaf:name
"Pele Gill"
sc
sc
bench:Article
Journal1
dc:creator
bench:Incollection
dc:title
swrc:
journal
"Journal 1 (1940)"
sc
bench:Book
dc:creator
:Frank_Ness
Article1
bench:MastersThesis
dc:title
dcterms:issued
foaf:name
"SPARQL"
"2006"
"Frank Ness"

Weitere Quellen zu RDF

- W3C RDF Primer

  http://www.w3.org/TR/rdf-primer/
- Vorlesungsfolien FGIS

  http://dbis/index.php?course=WS0708/Spezialvorlesung/Formale%20Grundlagen%20von%20Informationssystemen/folien.html

  (Vorlesungen 7.12.2007 und 12.12.2007)
- RDF Tutorial

  http://www.w3schools.com/rdf/default.asp
- Friend-of-a-Friend Projekt

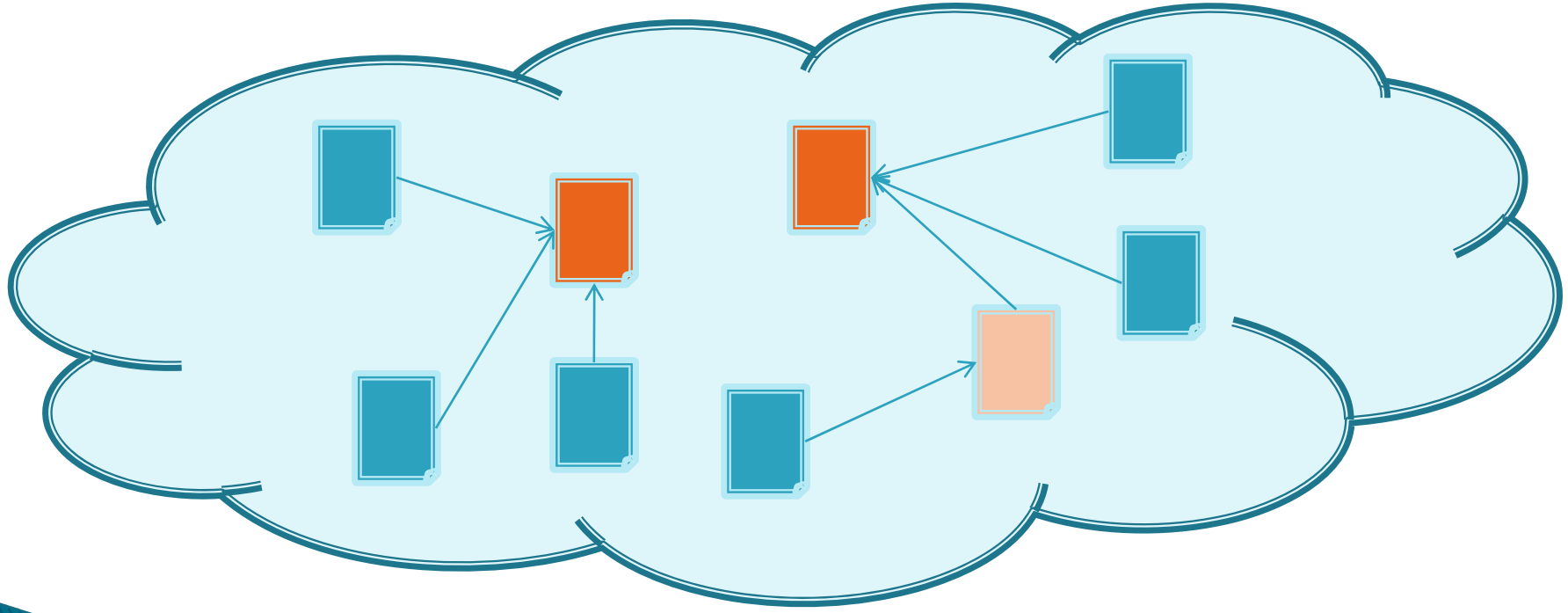
  http://www.foaf-project.org/

RDF ist ein wesentlicher Bestandteil des Team-Projekts und jeder sollte sich bis zur nächsten Woche entsprechend einarbeiten

PageRank

Hyperlink →

- Ranking-Schema zum Bewerten der Bedeutung von HTML-Seiten

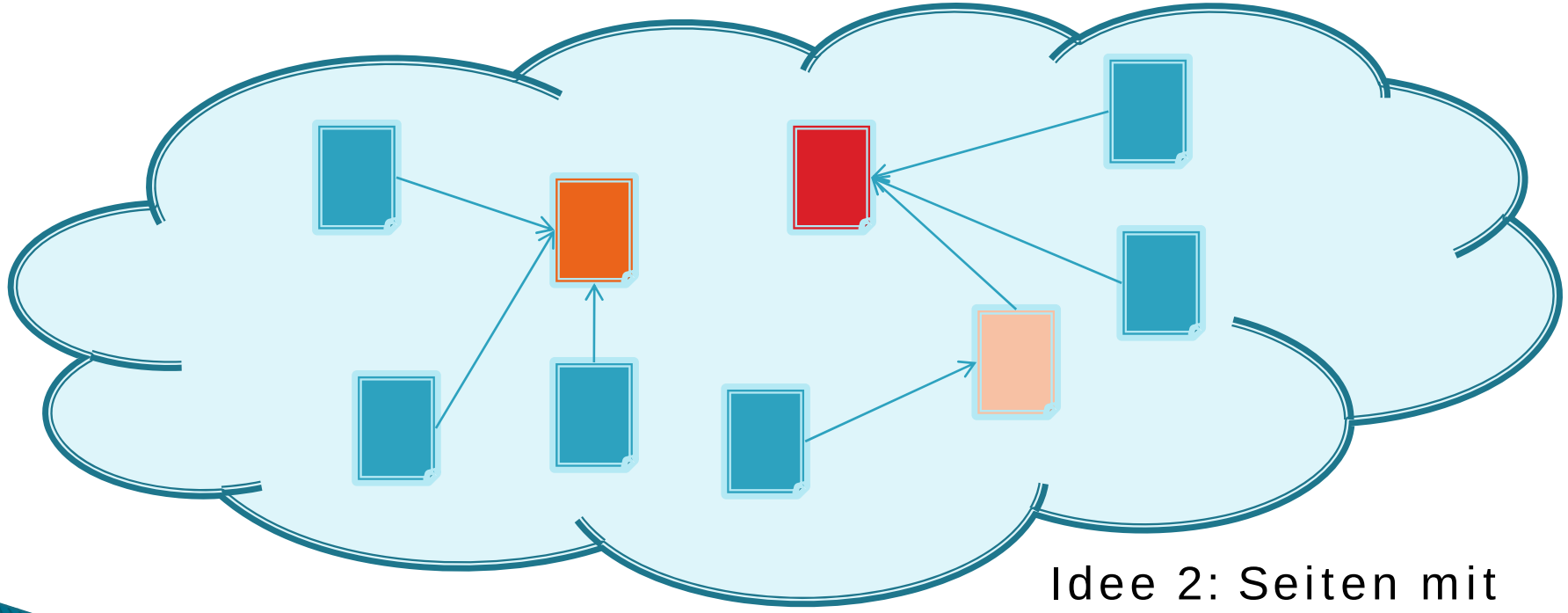


Idee 1: Seiten mit vielen eingehenden Links tendentiell wichtiger als andere!

PageRank

Hyperlink →

- Ranking-Schema zum Bewerten der Bedeutung von HTML-Seiten



Idee 2: Seiten mit eingehenden Links von wichtigen Seiten tendentiell wichtiger!

PageRank-Algorithmus

- Definition PageRank-Algorithmus PR
  - u, v: Web-Seiten
  - F_u: Menge aller Webseiten auf die u verlinkt
  - B_u: Menge aller Webseiten die auf u verlinken
  - $N_u := |F_u|$: Anzahl der Webseiten auf die u verlinkt
  - N: Anzahl aller Webseiten (sh. Folie 21)

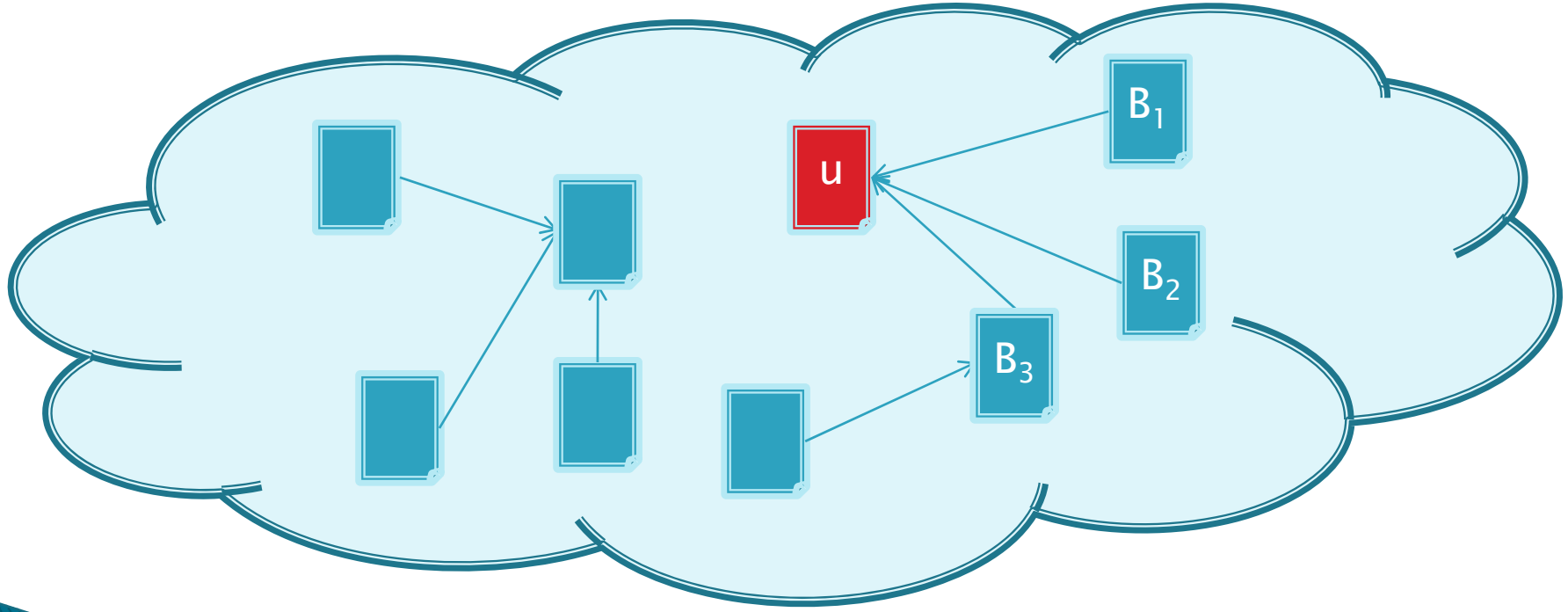
$$PR(u) := \sum_{v \in B_u} PR(v)/N_v$$

- Detaillierte Beschreibung

http://dbpubs.stanford.edu:8090/pub/showDoc.Fulltext?lang=en&doc=1999-66&format=pdf&compression=

Beispiel PageRank

$$PR(u) := \sum_{v \in B_u} PR(v)/N_v = PR(B_1) + PR(B_2) + PR(B_3)$$



$B_u = \{ B_1, B_2, B_3 \}$ mit $N_{B1} = N_{B2} = N_{B3} = 1$

PageRank-Erweiterung

- Erweiterung: Damping-Faktor

$$PR(u) := ((1-d)/N) + d * \sum_{v \in B_u} PR(v)/N_v$$

- Intuition: „Random Surfer Model“
  - d: Wahrscheinlichkeit dass Benutzer einem Link folgt; 1-d: Nutzer ist gelangweilt beginnt neue Surf-Session (z.B. Browser Bookmarks)
  - ((1-d)/N): Wahrscheinlichkeit dass Benutzer per Zufall auf u klickt (bei neuer Session)
  - $d * \sum_{v \in B_u} PR(v)/N_v$: Wahrscheinlichkeit dass ein Benutzer durch randomisiertes Surfen auf Seite u landet

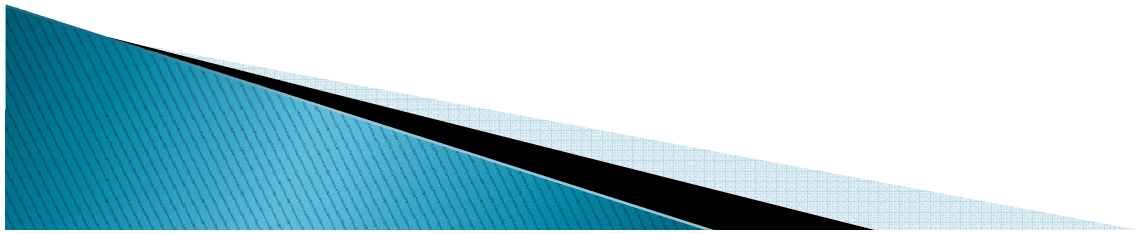
PageRank Lösungsverfahren

- Exakte Lösung schwierig und aufwendig, da rekursive Formel

  ➡ für Echtdaten (Millionen/Milliarden von RDF-Tripeln, d.h. Kanten im RDF Graphen) nicht realisierbar

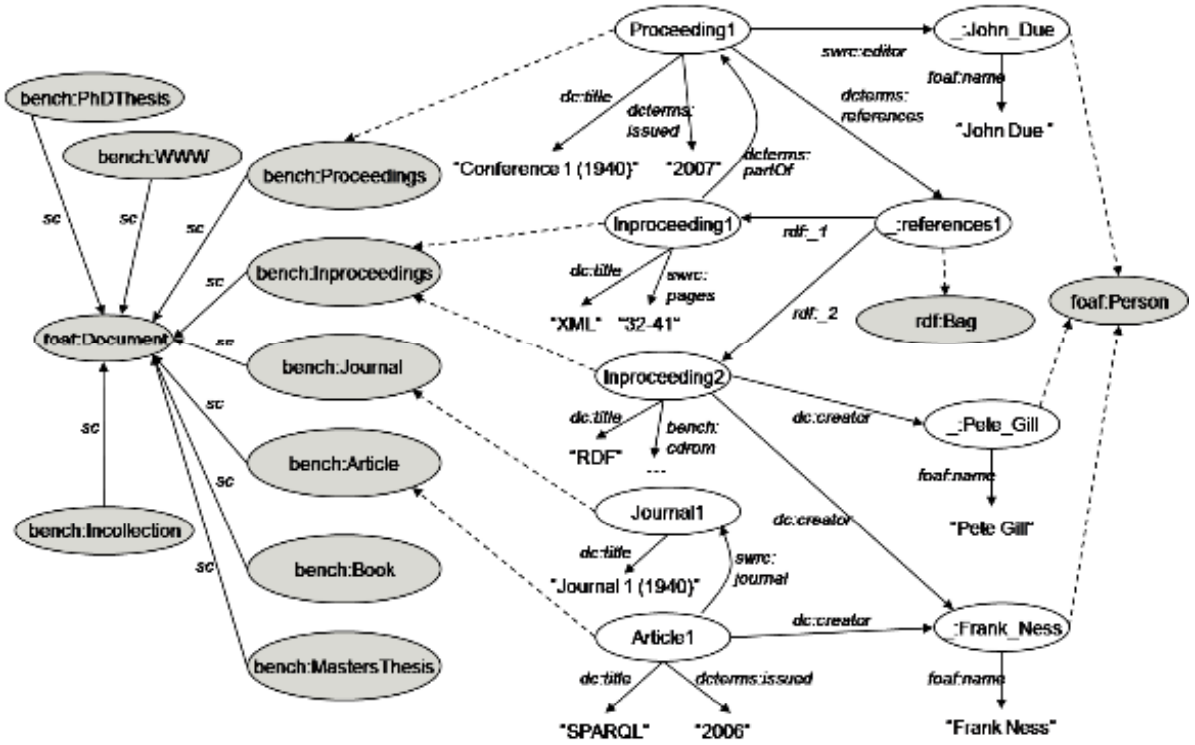
- Iteratives Lösungsverfahren liefert sehr gute Ergebnisse schon bei wenigen (<100) Iterationen; mehr Informationen unter

  http://pr.efactory.de/d-pagerank-algorithmus.shtml

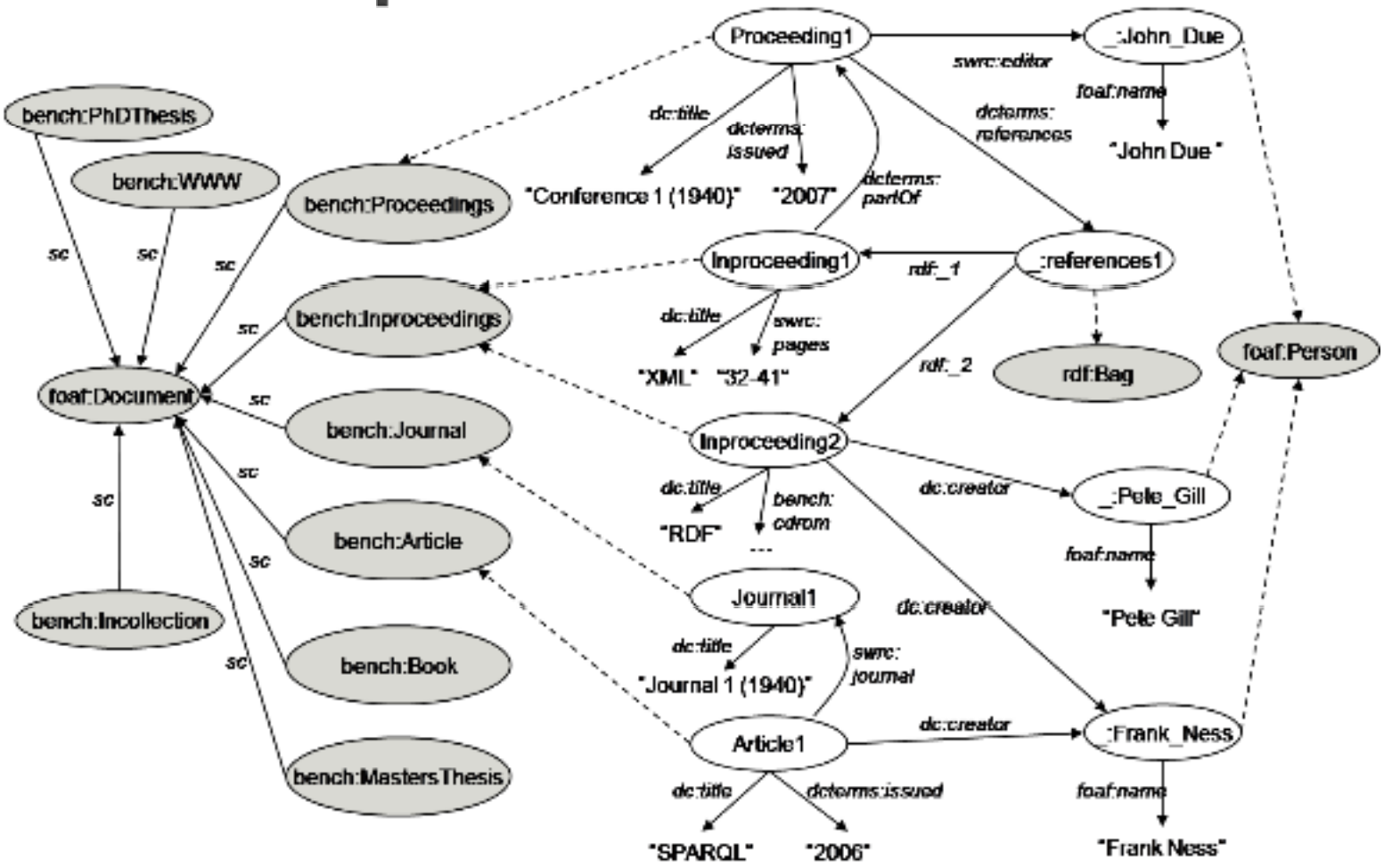


PageRank für RDF

- Interpretation von RDF-Graphen: Knoten als Webseiten, Prädikate als Links



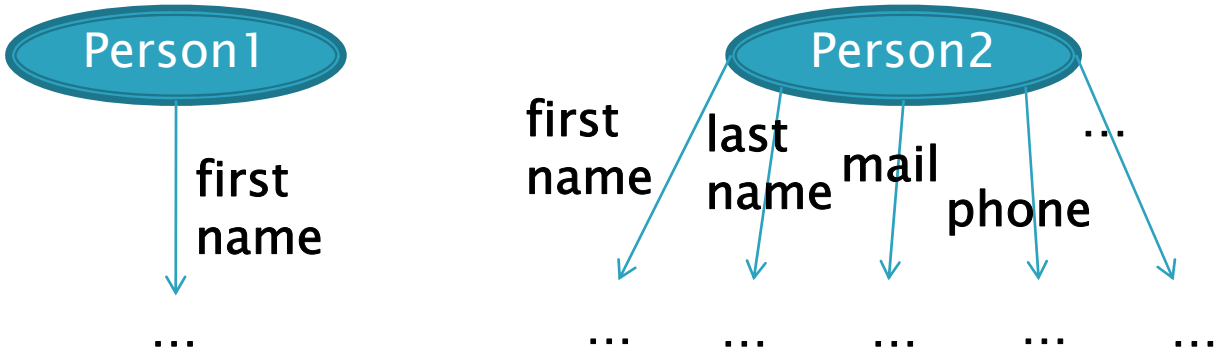
RDF Graph



Prädikate stellen Links dar.
URIs mit vielen eingehenden Links sind tendentiell wichtiger!

PageRank für RDF

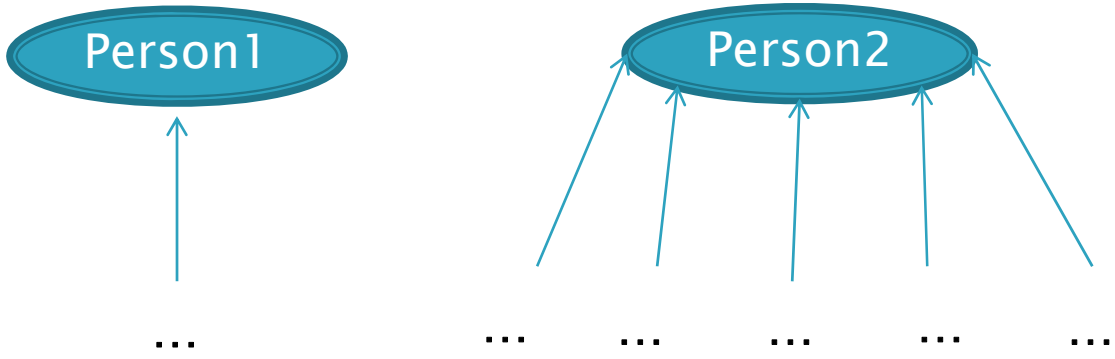
- Mit Hilfe vorheriger Beobachtung kann ein Ranking für Graphknoten unmittelbar mit dem Page-Rank Algorithmus berechnet werden
- Jedoch spielt bei RDF noch eine zweite Komponente eine Rolle: „Informationsgehalt“



Informationsgehalt von Person2 höher, daher tendentiell „wertvoller“ als Person1!

PageRank für RDF

- Informationsgehalt kann nach Inversion der Kanten mit dem PageRank Algorithmus (im Folgenden „PR^-“ genannt) berechnet werden:



- Gesamt-Ranking eines Knotens n könnte sich dann (mit Skalierfaktoren α und β) ergeben als

$$R(u) = \alpha * PR(u) + \beta * PR^-(u)$$

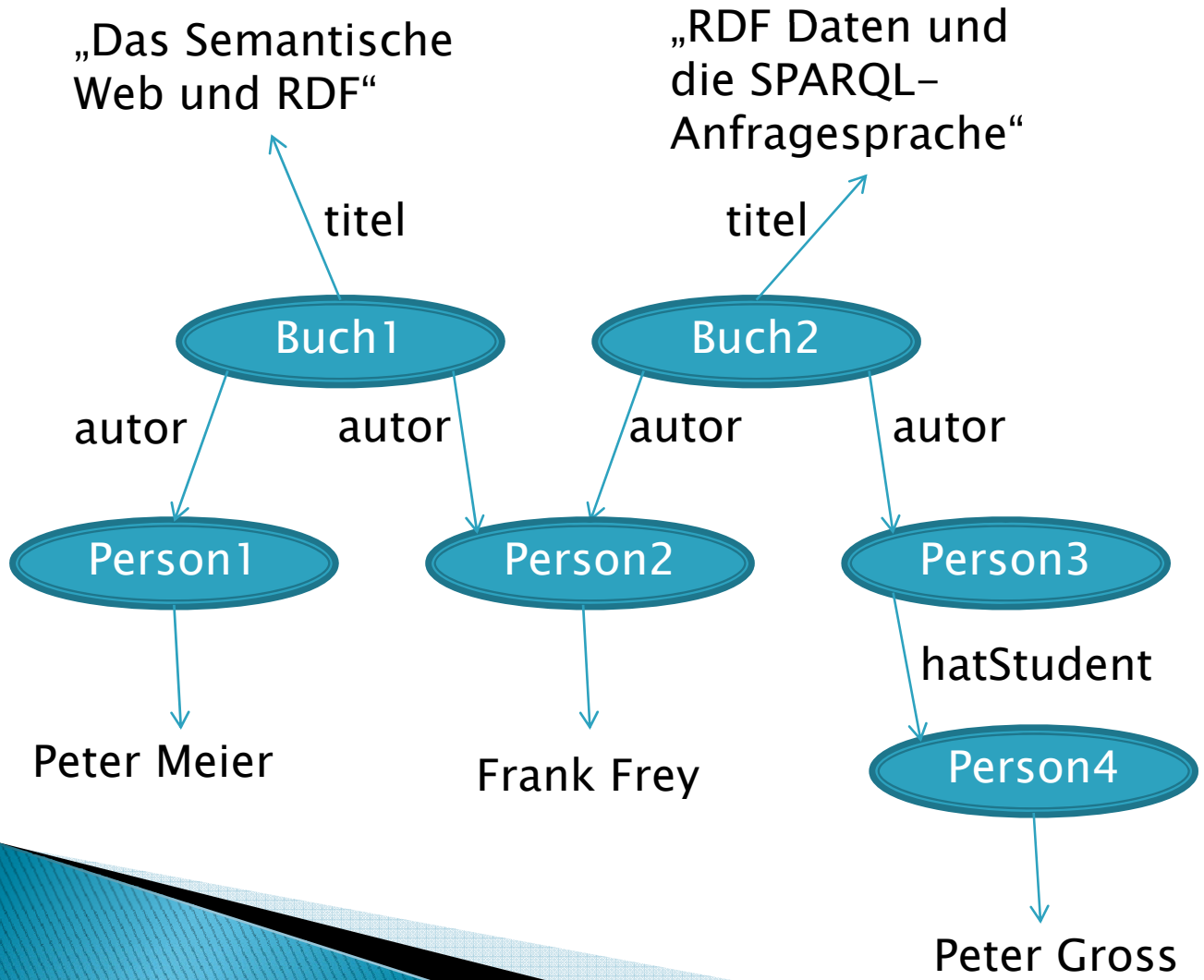
Keywort-Suche

- Single-Keywort Suche kann direkt auf den PageRanks der einzelnen RDF-Elemente aufbauen
- Ziel dieses Projekts: Suche nach mehreren Keywörtern gleichzeitig
- Siehe z.B.

  B. Aleman-Meza, C. Halaschek, I. B. Arpinar, and A. Sheth: Context-Aware Semantic Association Ranking
  In SWD Workshop, 2003.

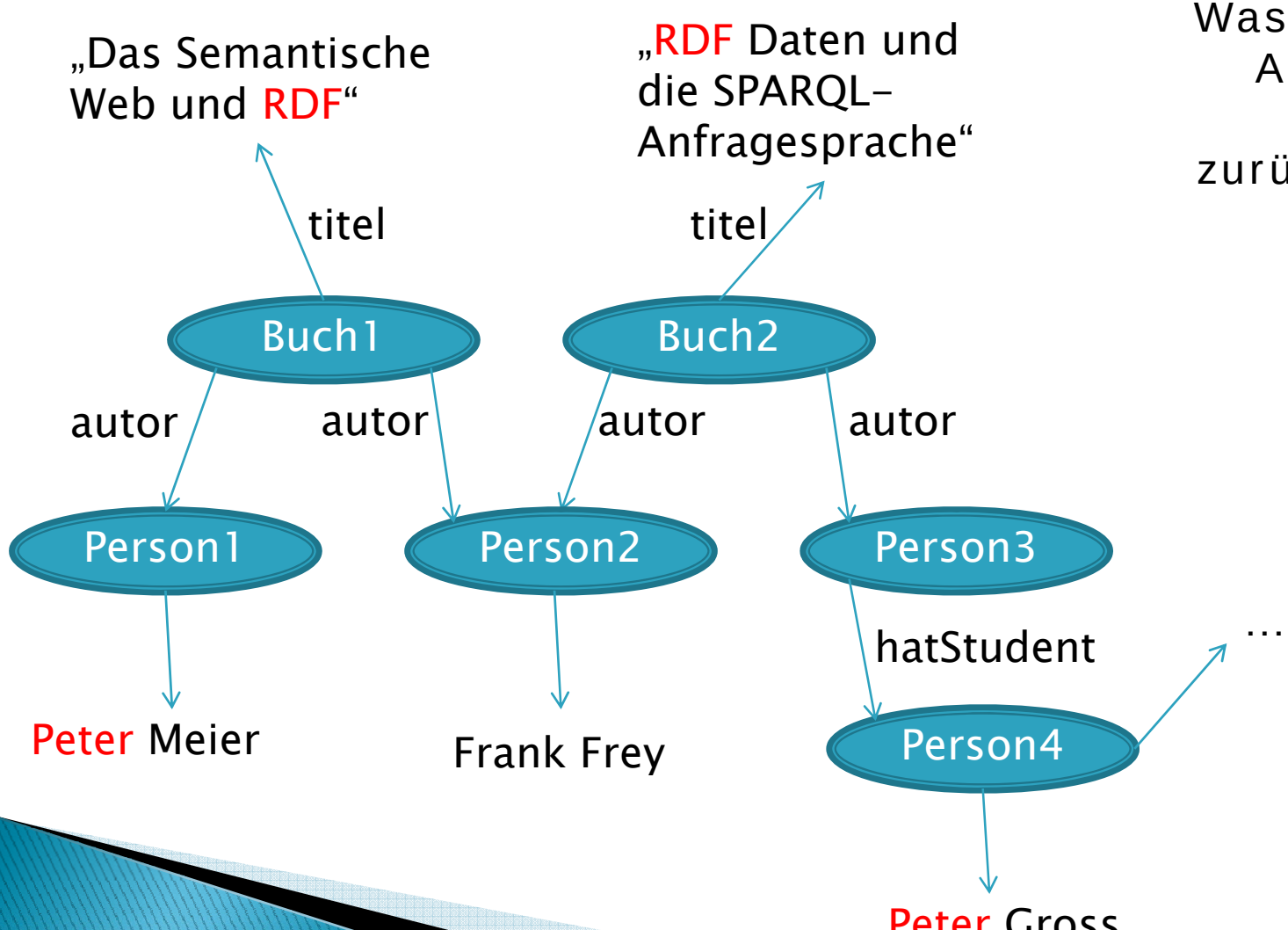
Aufgabe: eigenständiges Entwickeln von Lösungsstrategien für dieses Problem

Probleme bei Suche nach mehreren Keywörtern (Konjunktion)



Anfrage:
„Peter“ + „RDF“

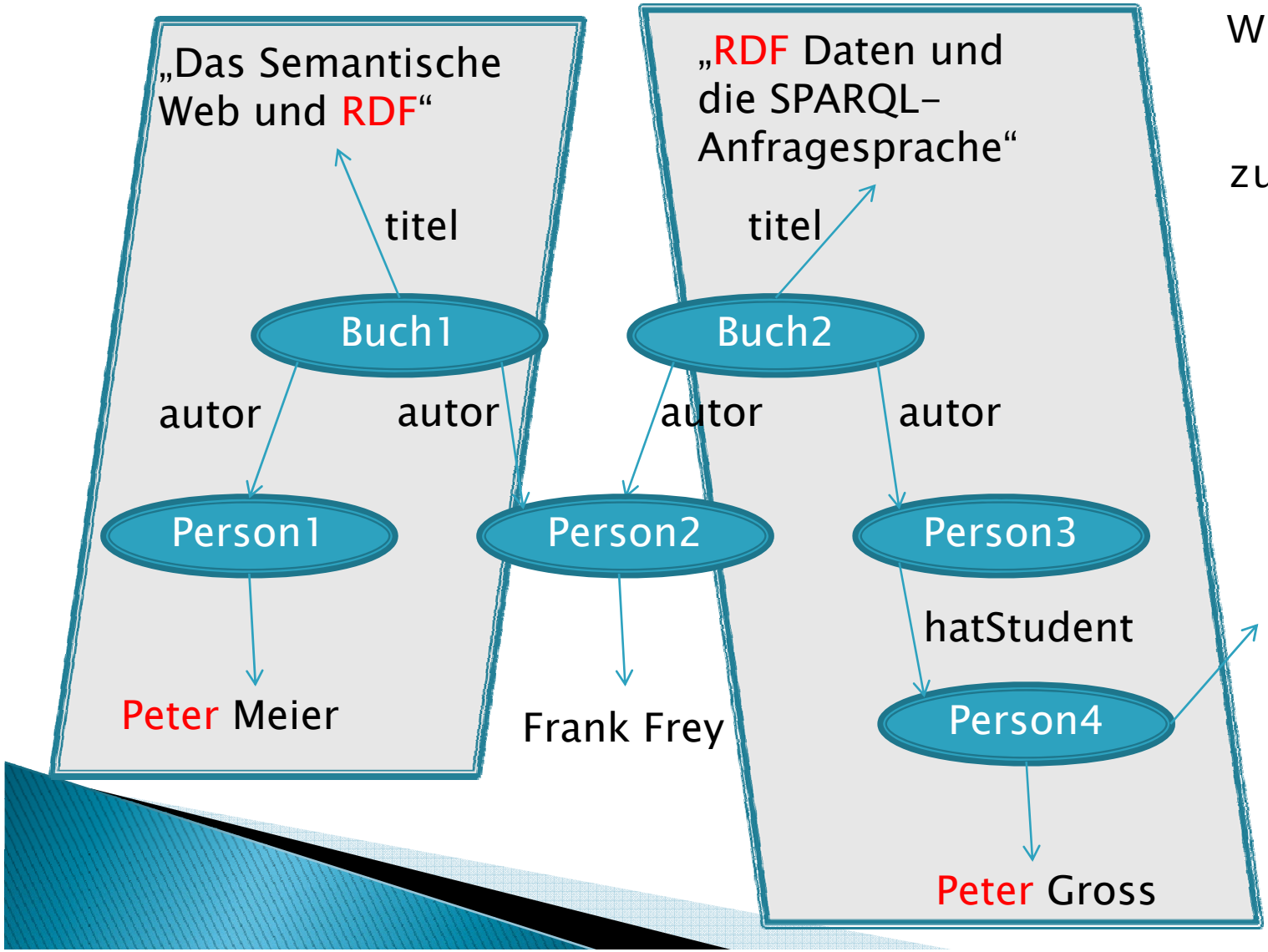
Probleme bei Suche nach mehreren Keywörtern (Konjunktion)



Was wird bei der Anfrage als Antwort zurückgegeben?

Anfrage:
„Peter“ + „RDF“

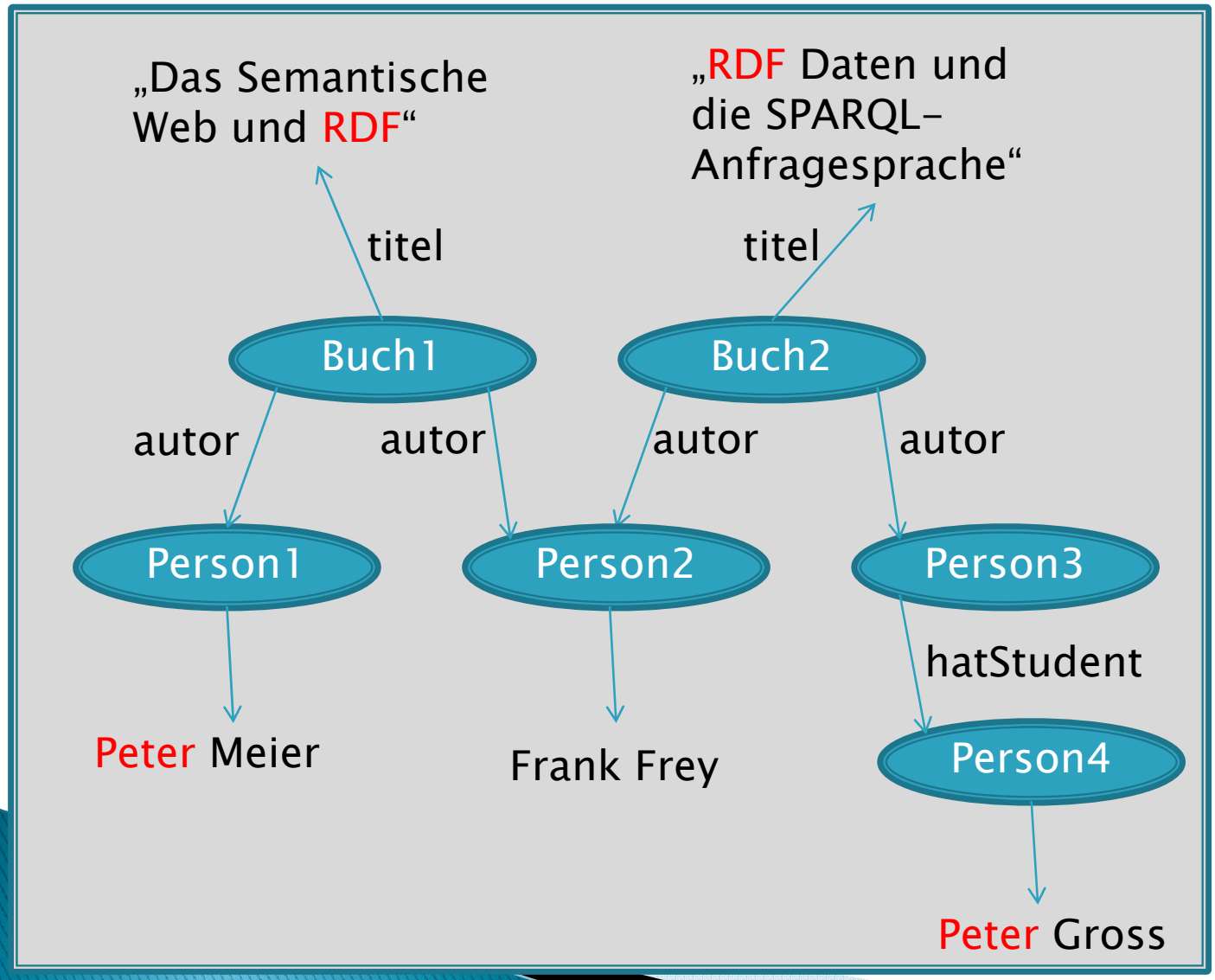
Probleme bei Suche nach mehreren Keywörtern (Konjunktion)



Was wird bei der Anfrage als Antwort zurückgegeben?

Anfrage:
„Peter“ + „RDF“

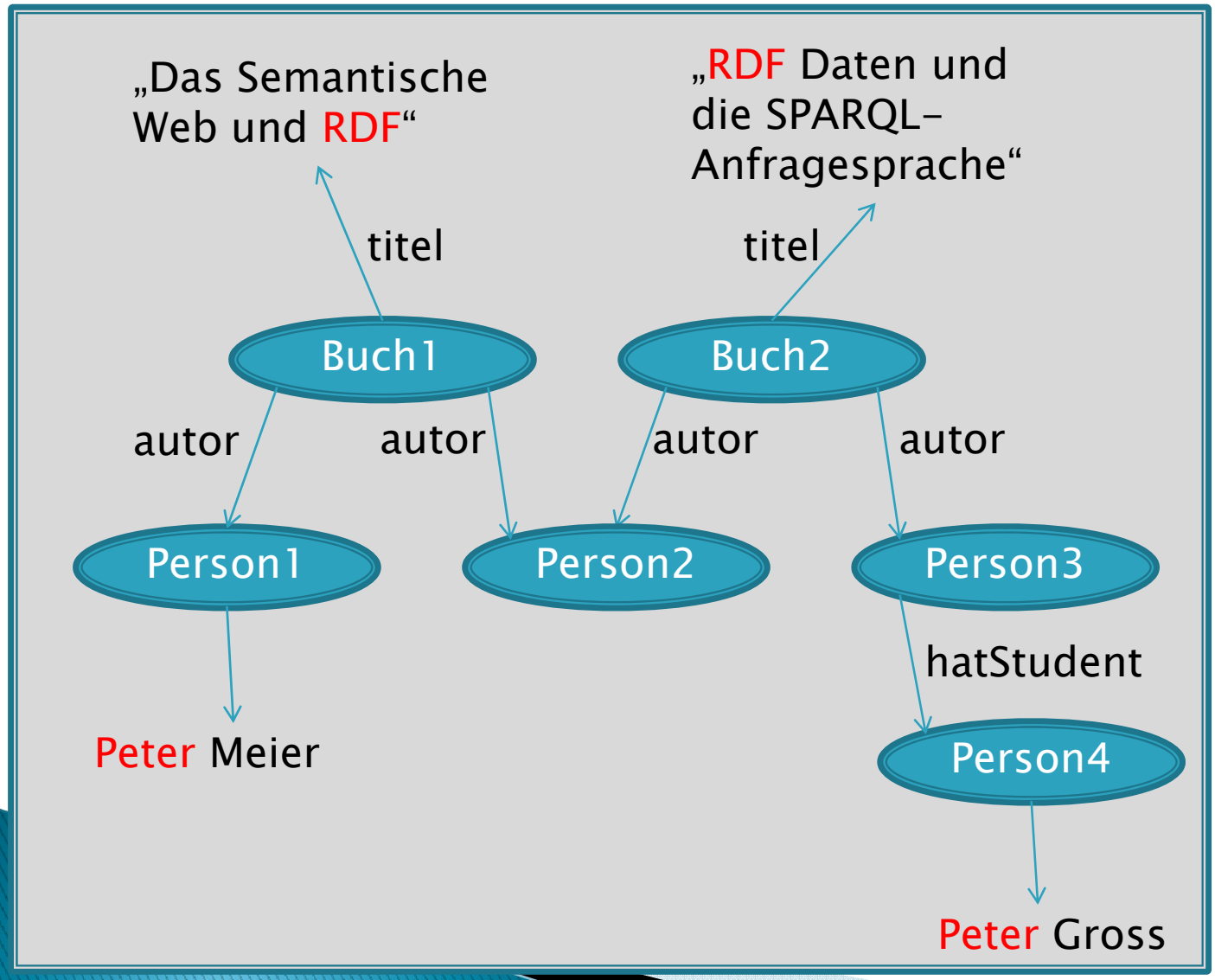
Probleme bei Suche nach mehreren Keywörtern (Konjunktion)



Was wird bei der Anfrage als Antwort zurückgegeben?

Anfrage:
„Peter“ + „RDF“

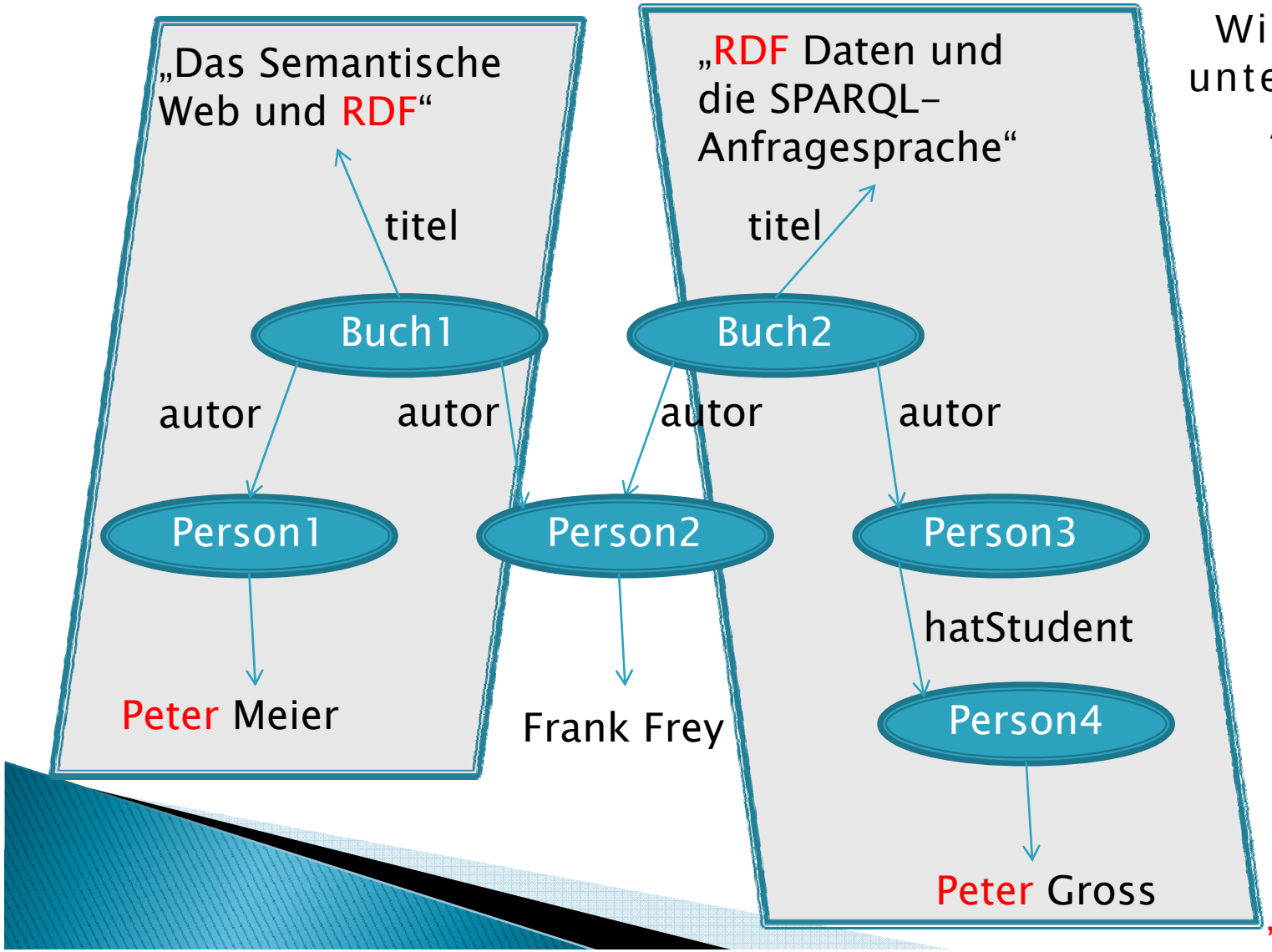
Probleme bei Suche nach mehreren Keywörtern (Konjunktion)



Was wird bei der Anfrage als Antwort zurückgegeben?

Anfrage:
„Peter“ + „RDF“

Probleme bei Suche nach mehreren Keywörtern (Konjunktion)



Wie werden die unterschiedlichen Antworten gerankt?

Anfrage:
„Peter“ + „RDF“

Bis nächste Woche…

- Einlesen in RDF
- Einlesen in andere Gebiete, je nach Zuständigkeit
- Interne Aufteilen der Team-Zuständigkeiten

- Roadmap
  - 28.10.: Vortrag über Crawler und Indexstrukturen zur effizienten Suche
  - 04.11.: Vorstellung der Konzepte beider Teams
  - 11.11.: Fertigstellung eines kompletten Entwurfs, Systemarchitektur etc.; Start der Implementierung