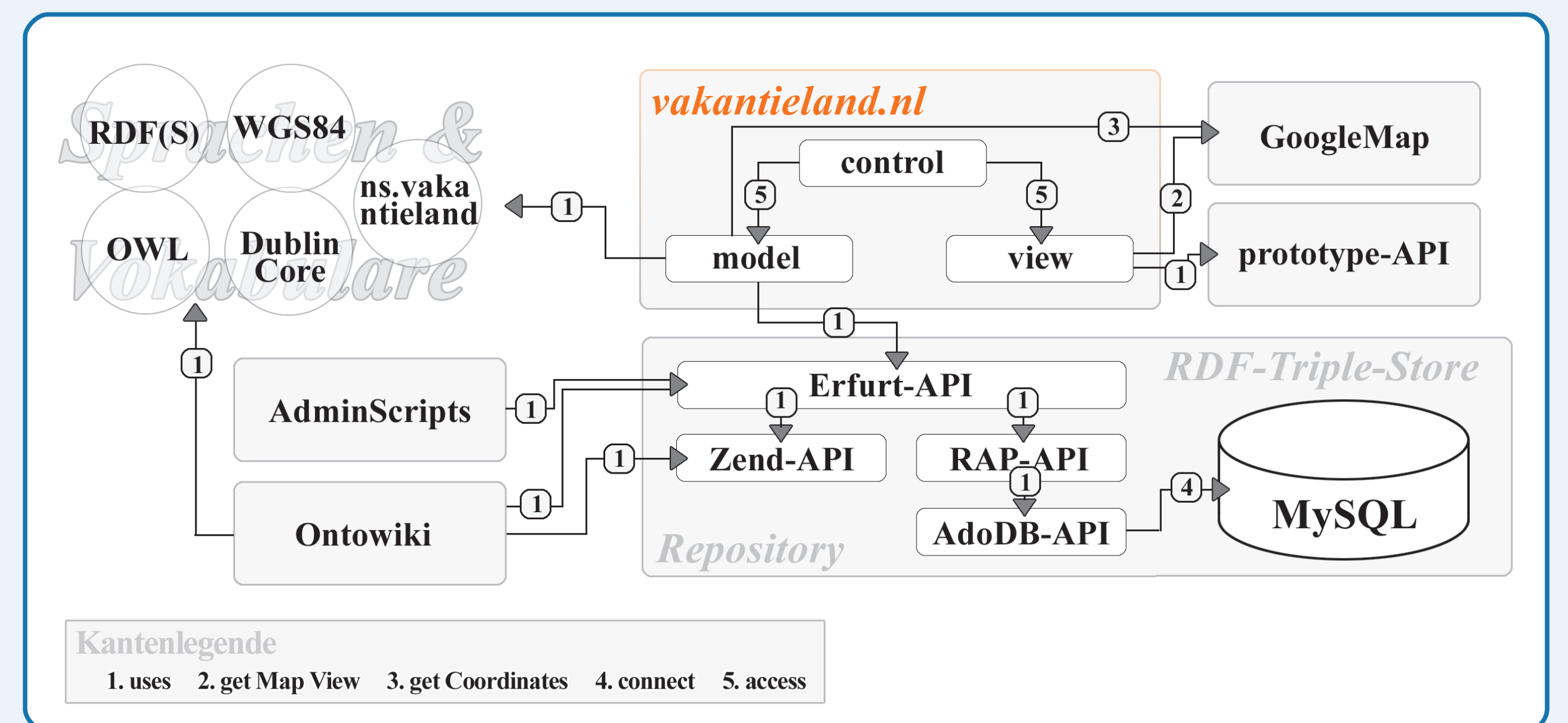


Xinnovations STI Berlin & CSW PhD Workshop

Performanzsteigerung datenbankgestützter RDF-Triple-Stores

MICHAEL MARTIN

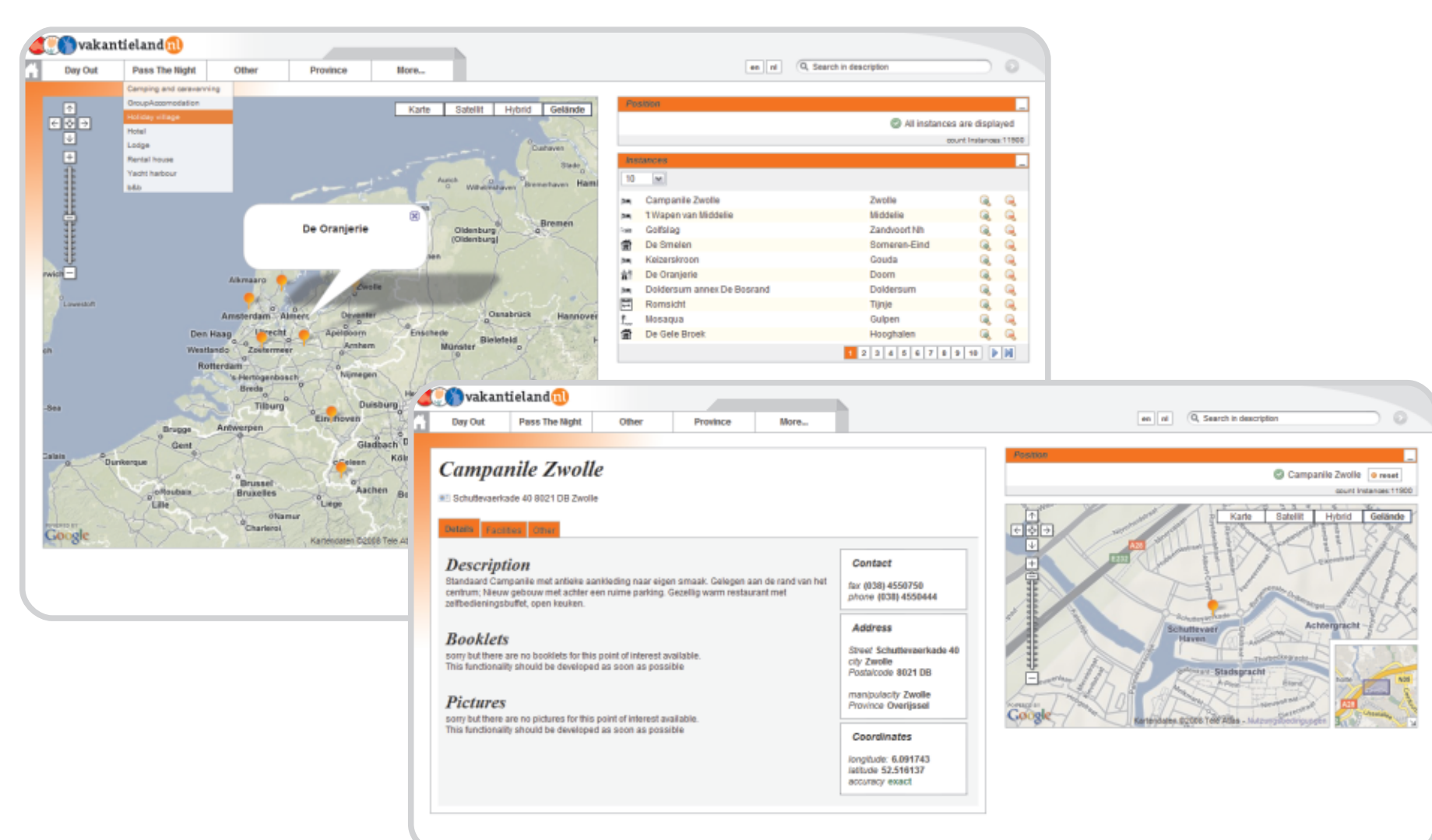
Die Benutzung von RDF-Triple-Stores im Semantic Web ist bei der Verwendung großer Wissensbasen fast unumgänglich. Hierzu hat sich unter anderem der Einsatz von Datenbankmanagementsystemen für grundlegende Speichermechanismen etabliert. Allerdings sind die Performanz- und Skalierbarkeitseigenschaften von RDF-Triple-Stores im Gegensatz zu herkömmlichen relationalen Datenbanken noch nicht optimal. Dies erschwert die Verbreitung von Technologien des Semantic Web sowie eine damit verbundene Entwicklung semantischer Web-Applikationen, bei denen der Fokus auf schnelle Datenverarbeitung gelegt wird. Mit diesem Forschungsvorhaben werden Ansätze zur Behebung dieses Problems untersucht.



Architektur des Anwendungsfalls vakantieland.nl mit RDF-Triple-Store

Ein motivierendes Beispiel

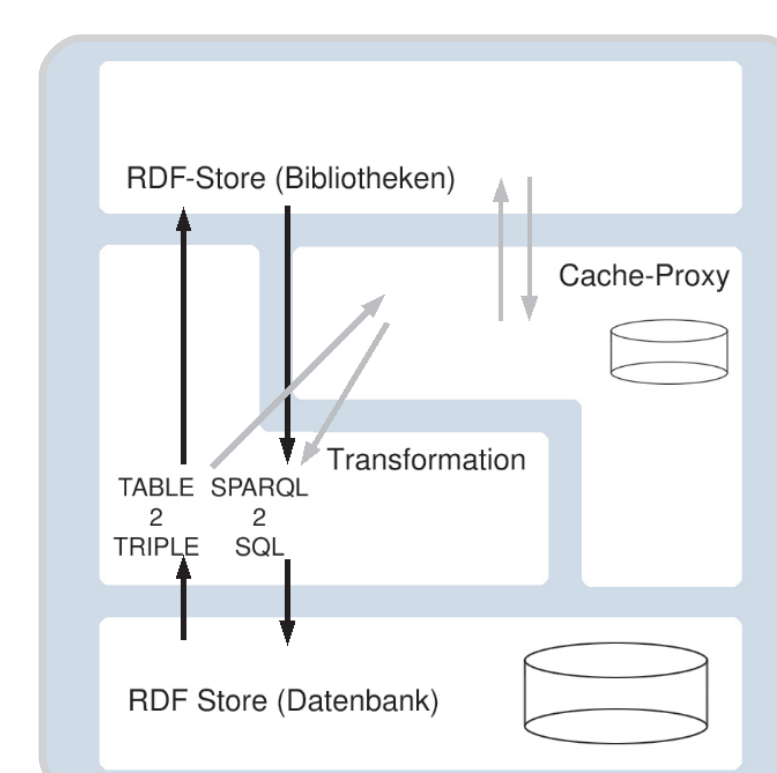
Während der Entwicklung der semantischen Web-Applikation vakantieland.nl, welche als Informationsplattform über touristische Ziele aus den Niederlanden entwickelt wurde, sind Performanzschwächen des zugrunde liegenden RDF-Triple-Stores (siehe Abb. oben) festgestellt worden.



Zur Behebung dieser Tatsache wurden auf den Anwendungsfall angepasste Caching-Algorithmen integriert. Zur Administration der Ontologie von vakantieland.nl wird das webbasierte Ontologie-Management-Werkzeug OntoWiki benutzt. Mittels zu erstellender Plugins könnte die anwendungsspezifische Caching-Lösung ebenfalls im OntoWiki zur Performanzsteigerung eingesetzt werden. Diese müsste allerdings bei Änderungen der jeweiligen Modellkomponente angepasst und erweitert werden, weshalb eine generische performanzsteigernde Lösung angestrebt wird.

Die Idee

Ein generischer Ansatz hierfür ist die Entwicklung eines Caches als Proxy-Schicht, welche im jeweiligen RDF-Triple-Store integriert werden kann. Die Interna dieser Proxy-Schicht sind noch weitestgehend zu evaluieren, werden sich aber insofern möglich an Optimierungsstrategien aus dem Datenbankbereich orientieren. Aufbauend auf einem naiven Caching-Algorithmus, dessen Performanzgewinne mittels adäquater Benchmarks gemessen werden, sind weitere diesen naiven Algorithmen zu integrieren.



Naiver Caching-Ansatz

| qid | S | P | O |
|-----|------|-----------|-------------|
| 1 | NULL | rdf:type | foaf:Person |
| 1 | NULL | foaf:name | NULL |

Query 1: SELECT ?x, ?y WHERE { ?x rdf:type foaf:Person. ?x foaf:name ?y }

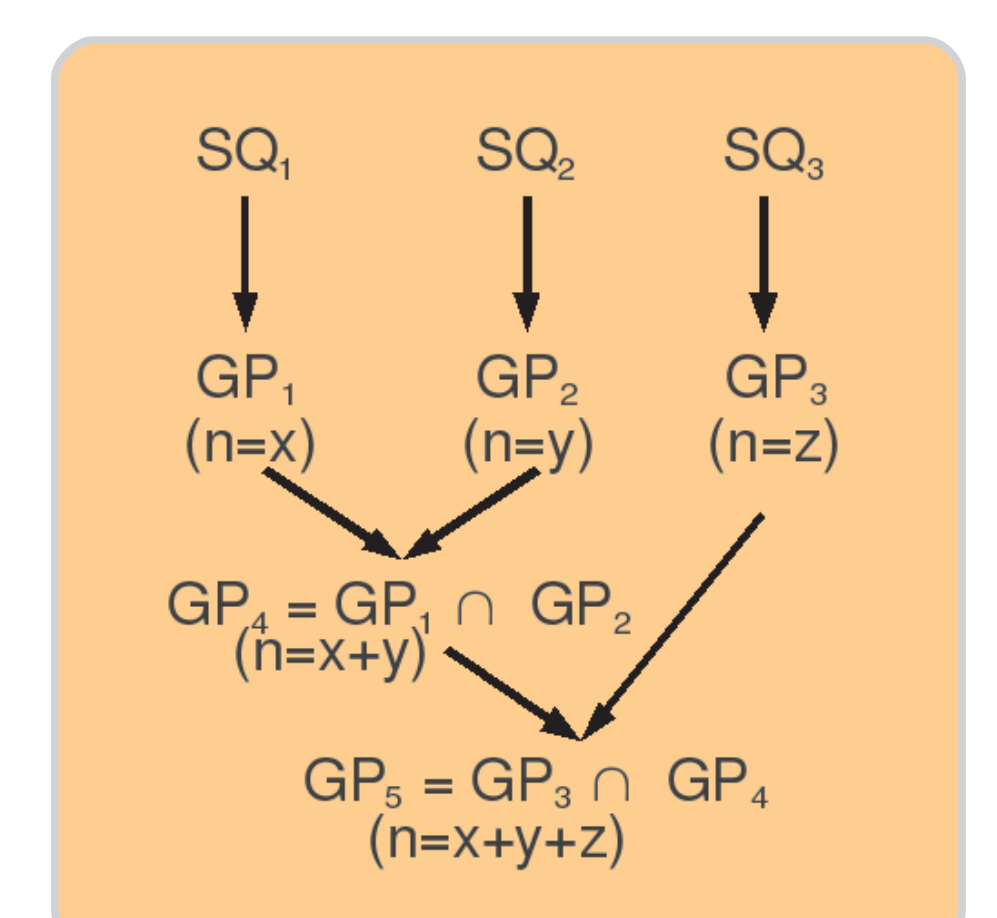
| qid | query | serial | count |
|-----|--------|-----------|-------|
| 1 | -hash- | <-result> | z |

Update: (Micha, rdf:type, foaf:person)
 Auswahl betroffener gecachter results:
 SELECT gid FROM graphPattern WHERE (S='Micha' OR S=NULL) AND (P='rdf:type' OR P=NULL) AND (O='foaf:person' OR O=NULL);

Objekten in Beziehung stehen, komplette Objektstrukturen in der Proxy-Schicht vorgehalten werden. Eine derartige Erweiterung würde auch die Eigenschaft von Web-Applikationen unterstützen, zumeist spezielle Sichten auf die in der Ontologie enthaltenen Informationen zu bieten.

Die Durchführung

Das avisierte Ergebnis des Forschungsvorhabens ist unter anderem die Erstellung eines abstrakten Modells zur Beschreibung der benötigten Eigenschaften und Algorithmen, die wiederum durch eine (Referenz-) Implementierung realisiert werden. Dieses Model wird hierbei die Elemente "Trippersistenz", "Anfragen", "Latenz" sowie "Ausführungs- und View-Materialisierungsgeschwindigkeit" beschreiben. Weiterhin werden Heuristiken für das View/Cache-Management sowie regelbasierte Mechanismen zur Query-Subsumption und Cache-Object-Invalidation erarbeitet. Gerade die Heuristiken werden zur Optimierung benötigt, um beispielsweise nur die meist frequentierten Daten zu materialisieren.



Eine Möglichkeit hierfür ist das Bilden von Schnittmengen über GraphPattern, die aus protokollierten SPARQL-Queries extrahiert wurden. Durch die Addition der Häufigkeiten lassen sich die meist benutzten Teile der GraphPattern ermitteln. Werden derartige Algorithmen in der Implementierung umgesetzt, müssen diese auf ihren Performanzgewinn hin durch Benchmarks validiert werden.

Info & Kontakt

Michael Martin
 martin@informatik.uni-leipzig.de

Dieses Forschungsvorhaben wird derzeit an der Universität Leipzig, Fakultät für Mathematik und Informatik, Institut für Informatik, Abteilung Betriebliche Informationssysteme bearbeitet.

Informationen zum aktuellen Stand der Arbeit werden später auf der Website der Webseite der Forschungsgruppe AKSW (<http://aksw.org>) veröffentlicht.