Xodx Simulationsarchitektur

Überwachungs- und Statistik-Komponente

Franz Teichmann

Universität Leipzig

11. Januar 2016

Inhalt

Semantic Web

- Erweiterung des aktuellen Web, um neue Anwendungsfelder zu erschließen und Arbeit zu vereinfachen
- Interoperabilität von Domänenwissen über offene Wissensbasen
- Verwendung hochstrukturierter Daten im Gegensatz zu unstrukturierten oder semi-strukturierten Daten im herkömmlichen Web
- Dadurch bessere maschinelle Verarbeitung: Darstellung der Semantik der Daten
- Knowledge Engineering
- Verwendung eines einheitlichen Frameworks: RDF

Triplestores

- Speziell für die Verwendung im Semantic Web optimierte Datenbank für Triple bzw. Quadrupel
- Vernetzte semantische Abfrage über SPARQL möglich
  - Sparql 1.1 umfasst u.A. select, describe, ask, update queries und Aggregationsfunktionen
- OpenLink Virtuoso
  - Open-Source Variante des Virtuoso Universal Server
  - Leistungsfähiger Triplestore für diverse Plattformen
  - Unterstützt SPARQL 1.1 über Schnittstelle zur Datenbank (SPARQL-Endpoint)
- Apache Jena als In-Memory-Store
  - Open Source Framework zur Entwicklung von Semantic Web Anwendungen in Java
  - Verschiedene Nebenprojekte mit Schnittstellen zu diversen Datenquellen (z.B. Fuseki Servlet als http-Interface)
  - Möglichkeit, model in Arbeitsspeicher einzulesen und mit Sparql 1.1 abzufragen

Vokabulare

- Hilfsmittel für die Erstellung von Wissensbasen
- Ontologien mit großem Verbreitungsgrad
- Geben Prädikate vor, um domänenübergreifende Semantik darzustellen
- Beispiel foaf
  - Friend of a friend: Ontologie zur Beschreibung von Personen und Beziehungen
  - Beschreibung von sozialen Netzwerken
  - foaf:Person, foaf:name, foaf:box, foaf:knows

Xodx als DSSN

- Xodx umfasst eine Implementation von Basisfunktionalitäten im DSSN
  - Distributed
  - Semantic
  - Social Network
  - Umfasst Registrierung, Versenden von Nachrichten/Links/Bildern, Friending
- Entwickelt unter GPL2
- Umsetzung unter Verwendung von Erfurt, PubSubHubbub, Semantic Pingback
  - Erfurt: Semantic Web Api für PHP5, Verwendung von Zend
  - Saft: aktuelles Projekt in Entwicklung, verbesserte innere Architektur
- Verwendung von Zend
  - Open Source, objektorientiertes Framework für PHP, Entwicklung komplexer Webanwendungen

Ziele der Simulation

- Prüfung auf Bugs oder Fehlverhalten durch umfangreiche Simulation mit Realdaten
- Verwendung von Docker-Images, um Knoten im Netzwerk zu simulieren
- Nutzung eines Twitterkorpus und Replay-Agenten
- Dateneffizienz
  - Messen der Anzahl an gespeicherten Triple
  - Proportionales Wachstum erwartet
- Verlorene Nachrichten
  - Nutzer folgen anderen Nutzern und erhalten deren Posts
  - Followers oder deren Anzahl dem einzelnen Nutzer nicht bekannt
  - Anzahl an abgefragten Nachrichten muss im Gesamtüberblick inferiert werden
- Performanceprobleme
  - Messen der Zugriffszeit auf den eigenen Knoten über http-request
- Verwendung von Linked Data Technologien für die Statistik

Datacube Vokabular

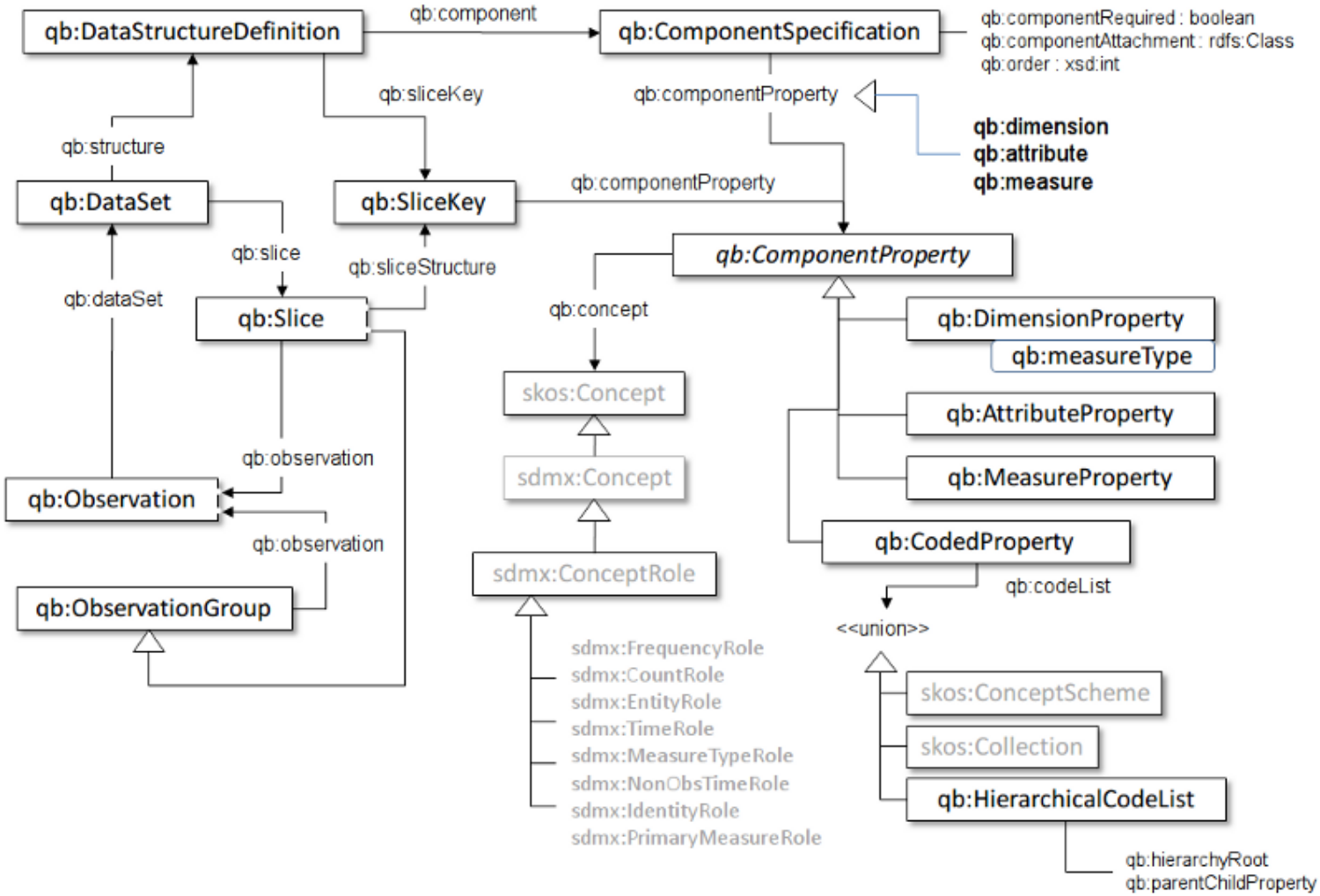

Abbildung: Outline des Datacube Vokabulars

Xodx-Architektur

- Verwendung von MVC nach Zend (Zend_Application)
- Bootstrapping-Funktion ausgehend von einer Indexdatei
- Controller
  - Hauptfokus der Arbeit
  - direkt über http-requests erreichbar (Action Methoden über Zend aufgerufen)
- Model
  - Repräsentieren das Datenmodell, Verbindung zum Triplestore erlaubt ausführen von update, select und ask-queries
- View
  - weniger Teil der Arbeit
  - funktioniert über ein Template mit content-Datenstruktur

Statistik-Controller

- Implementierung eines zusätzlichen Controllers für Xodx
- Über http-requests erreichbar, arbeitet mit dem model, um Observations zu generieren
  - Daher Planung, den Controller in Pausenzeiten während der Simulation aufzurufen
  - Kein unfriending, bei neuem Follower werden alle Posts ,,nachgereicht"
- Methoden
  - getStatsAction: Hauptmethode, sammelt zentral Beobachtungen der private-Methoden
  - readStoreAction: ermöglicht Analyse, indem alle Triple aus dem Store zu html-Tabelle geparst werden

Auswertung des Datacubes

1. Messung zu verschiedenen Zeitpunkten auf jedem Agenten des Netzes, Speicherung in Datei
2. Zusammenführen des Datacubes, auch Zwischenergebnisse möglich
   - Statischer Teil: Dataset- und Datastructure Definitionen
   - Dynamischer Teil: Observations
   - Übersetzung nach rdfxml mit rapper notwendig
3. Prüfung auf Validität: Java Application
   - z.B. Kollision von Identifiern (sehr unwahrscheinlich)
4. Auswertung, verschiedene Möglichkeiten
   - In-Memory Store mit Apache Jena Java executable archives und Sparql-Queries
   - Einlesen in Cubeviz, Erstellung von Visualisierungen für Messwertegraphen

Arbeitsplan

1. Verknüpfung mit Replay-Agent
2. Ausführen der Testläufe in Abhängigkeit von den Twitterdaten, vllt. mehrere Läufe
3. Zwischenzeit: Ausarbeitung und Test zusätzlicher Queries zur Auswertung
4. Zusammenfügen des Datacubes
5. Auswertung der Simulation in Seminararbeit

Live-Demo aktueller Stand

1. Xodx Look and Feel
2. Statistik Kontroller
3. Erstellung von Observations
4. Zusammenführen des Cubes
5. Validierung
6. Auswertung: verlorene Nachrichten