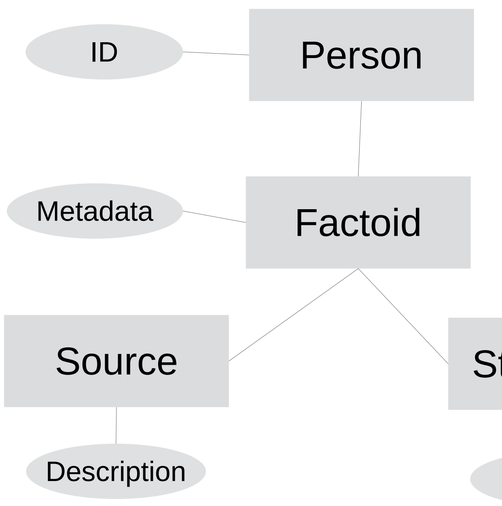
Prosopographische Interoperabilität (IPIF)
Stand der Entwicklungen

Georg Vogeler (georg.vogeler)[1], Richard Hadden (richard.hadden)[2], Matthias Schlögl (matthias.schloegl)[2], Gunter Vasold (gunter.vasold)[1]
[1]Universität Graz, @uni-graz.at [2]Österreichische Akademie der Wissenschaften, @oeaw.ac.at

Prinzip: einfache API



IPIF will **nicht** die Ausdrucksmächtigkeit von RDF basierten Modellen erreichen oder sich vollständig auf Upper-Level-Ontologies wie CIDOC-CRM abbilden lassen.

Einfache Filter: `name`, `place`, `relatedTo`, `from`, `to`, `memberOf` ...
`/statements?name=John%20Smith` (Alle statements, die den Namen „John Smith“ enthalten)
`/statements?place=Graz&from=1900&to=1910` (Alle statements, die einen Ortsangabe Graz haben und in den Zeitraum 1900–1910 fallen)
Als “short-cuts” zu informationen von anderen Endpoints:
`/person?name=John%20Smith` (Personen, über die es ein Statement mit dem Namen “John Smith” gibt)

… doch nicht so einfach? ...

Kombinierte Filter:
`https://example.org/ipif/v0.1/person/?name=Georg&place=Graz&from=2010&to=2015`

Autocomplete

Implementation

https://gitlab.com/richardhadden/pysolaar
https://gitlab.com/acdh-oeaw/apis/apis-ipif-solr

Effiziente Filter

Bevorzuge Filter, die einen kleinen Datenausschnitt erzeugen, denn es ist einfacher, mehrere API-Aufrufe clientseitig zu kombinieren als ein zu großes serverseitiges Ergebnis clientseitig einzuschränken.

Labels für abstrakte Entitäten

Keine URI ohne Label: abstrakte Konstrukte sollten immer eine menschenlesbare, semantisch besetzte Alternative besitzen, deren mangelnde semantische Präzision (es können fehlerhafte, veraltete oder umstrittene Angaben sein) hingenommen werden muss.

Implementation

Erlaube Implementationen, die die Definitionen der API performant umsetzen, auch wenn sie nicht explizit das von der API verwendete konzeptuelle Datenmodell realisieren.

https://prosopography.org