Distributed Vocabulary Development with Version Control Systems

Lavdim Halilaj, Steffen Lohmann, Christian Mader, Sören Auer

University of Bonn / Fraunhofer IAIS

Smart Descriptions & Smarter Vocabularies (SDSVoc)
Amsterdam, November 30th and December 1st, 2016

Motivation

General GitHub statistics (April 2016):

- > 35 million repositories
- > 14 million users

Well-known ontologies/vocabularies:

- Schema.org
- Description of a Project (DOAP)
- Friend-of-a-Friend (FOAF)
- The Music Ontology
- Edamontology (EDAM)
- Human Disease Ontology
- …

Our vocabularies:

- Mobivoc
- ScorVoc
- AutomationML
- LiDaKra
- Oddete
- OpenBudget
- ….

Distributed Vocabulary Development

Collaboration Support:

- *Governance*: roles, permissions, etc.
- *Communication*: issue tracking, notifications, etc.
- *Provenance*: revision history, semantic diffs, etc.

User Experience:

- *Documentation*: generated HTML, etc.
- *Visualization*: node-link diagram, etc.
- *Editor agnostic*: serializations, normalizations, etc.

Quality Assurance:

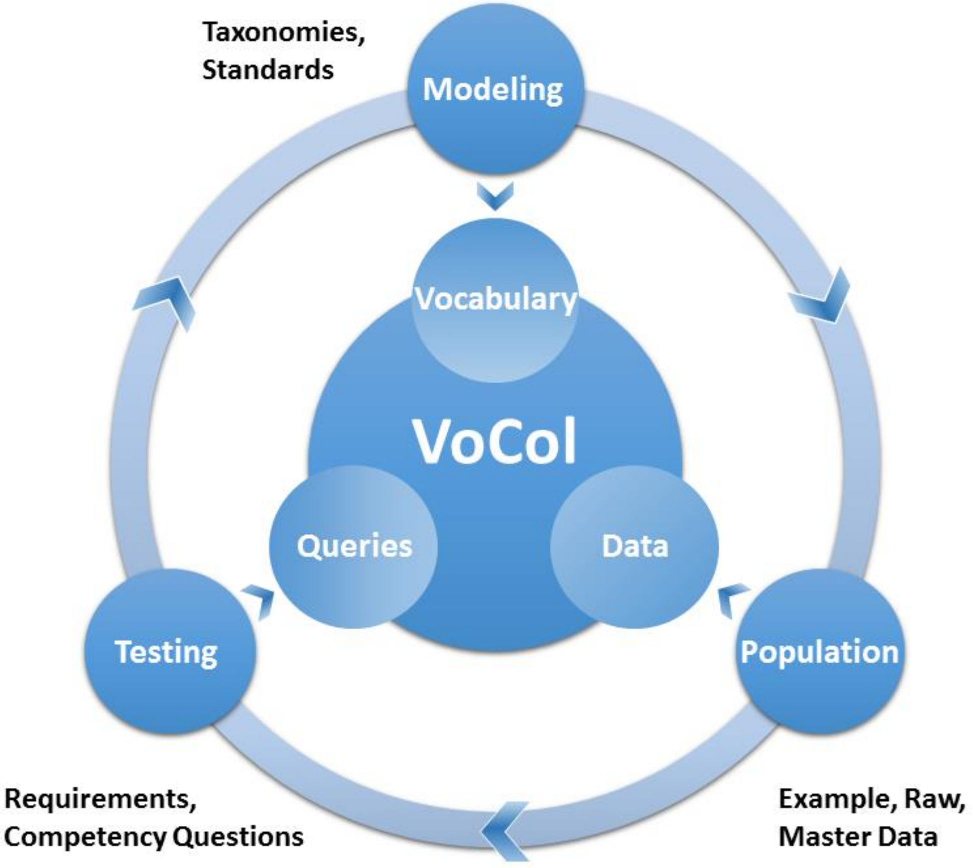
- *Syntactic Validation*: RDF/OWL compliance, etc.
- *Semantic Validation*: consistency checking, etc.
- *Testing*: competency questions, etc.

Vocabulary Deployment:

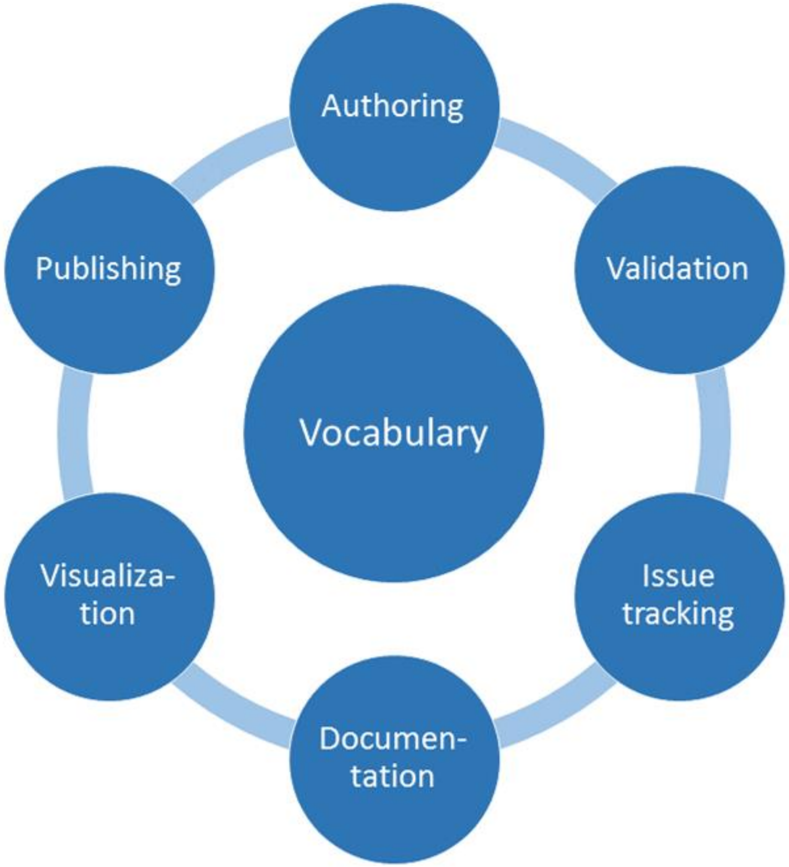
- *Machine accessibility*: content negotiation, etc.
- *Internationalization*: multilinguality, etc.
- *Querying*: SPARQL endpoint. etc.

→ Partly well-covered by Git-based version control + repository hosting platforms (e.g. GitHub) [Git4Voc16]

Distributed Vocabulary Development

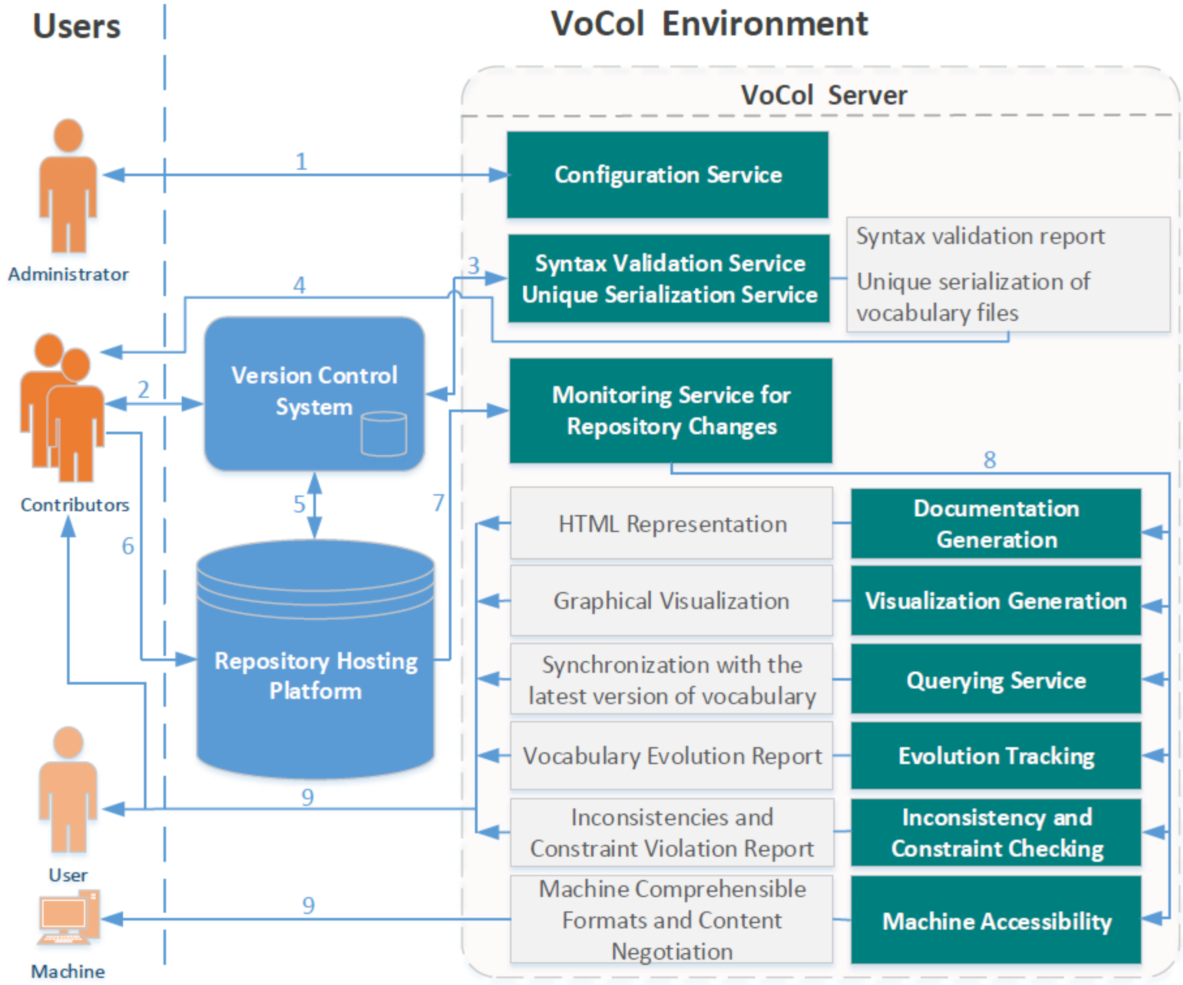


Round-trip development



Vocabulary development lifecycle

VoCol Architecture and Workflow



VoCol Server

Configuration Service

Syntax Validation Service
Unique Serialization Service

Syntax validation report

Unique serialization of vocabulary files

Version Control System

Monitoring Service for Repository Changes

Repository Hosting Platform

HTML Representation — Documentation Generation

Graphical Visualization — Visualization Generation

Synchronization with the latest version of vocabulary — Querying Service

Vocabulary Evolution Report — Evolution Tracking

Inconsistencies and Constraint Violation Report — Inconsistency and Constraint Checking

Machine Comprehensible Formats and Content Negotiation — Machine Accessibility

Legend

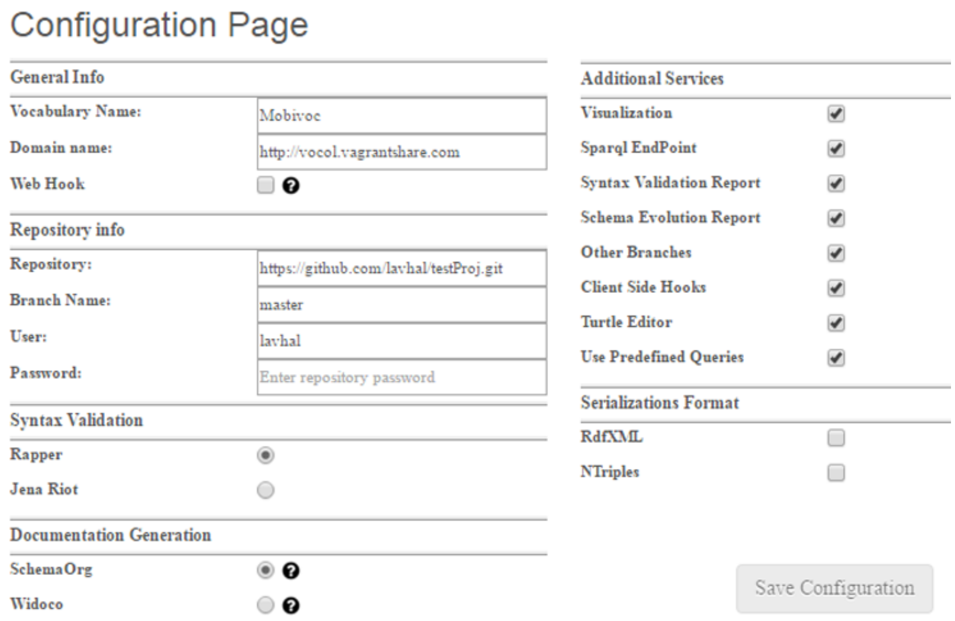
🟩🟦 Component or Service

⬜ Result or Artifact

1. Configure VoCol Environment
2. Commit local changes
3. Check for syntactic errors and serialize the vocabulary files according to a unique format
4. Deliver the validation report and the serialized vocabulary files
   - In case of errors, stop the commit and prevent users from pushing to the repository
   - Otherwise, commit succeeds
5. Push changes to the repository
6. Make changes using web interface of the repository service
7. Send notification with changes to the VoCol Server
8. Trigger following actions:
   - Generate documentation
   - Generate visualization
   - Synchronize the SPARQL endpoint
   - Generate the evolution report
   - Generate the inconsistency and constraint violation report
   - Serialize the vocabulary in different formats and activate content negotiation
9. Deliver the generated artifacts

→ Loose coupling, webhook mechanism, encapsulation via Vagrant & Docker

VoCol: Configuration

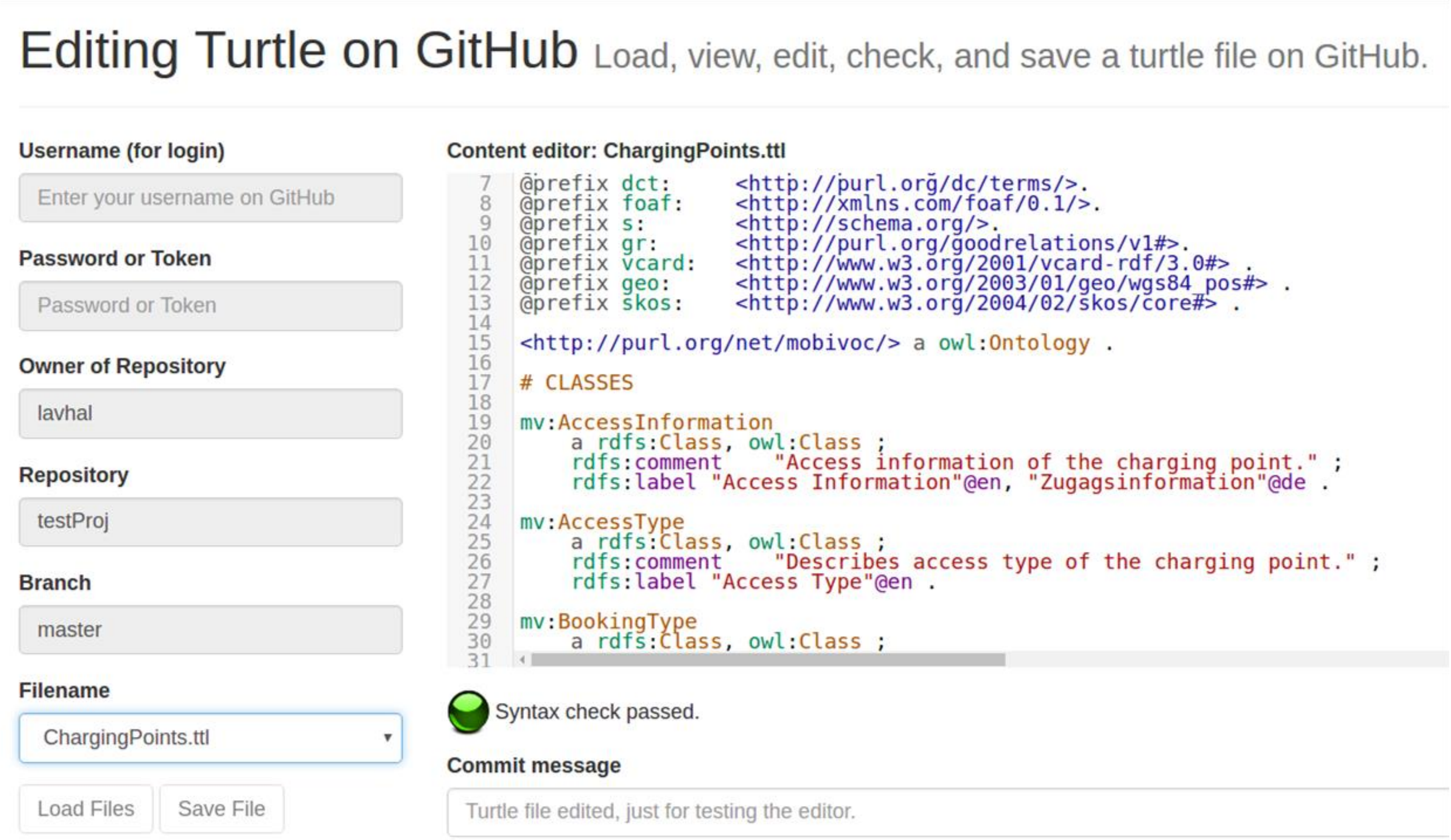


Configuration Page

General Info	
Vocabulary Name:	Mobivoc
Domain name:	http://vocol.vagrantshare.com
Web Hook	□ ?
Repository info	
Repository:	https://github.com/lavhal/testProj.git
Branch Name:	master
User:	lavhal
Password:	Enter repository password
Syntax Validation	
Rapper	●
Jena Riot	○
Documentation Generation	
SchemaOrg	● ?
Widoco	○ ?

Additional Services	
Visualization	☑
Sparql EndPoint	☑
Syntax Validation Report	☑
Schema Evolution Report	☑
Other Branches	☑
Client Side Hooks	☑
Turtle Editor	☑
Use Predefined Queries	☑
Serializations Format	
RdfXML	□
NTriples	□

VoCol: Integrated Editing



Editing Turtle on GitHub Load, view, edit, check, and save a turtle file on GitHub.

Username (for login)

Enter your username on GitHub

Password or Token

Password or Token

Owner of Repository

lavhal

Repository

testProj

Branch

master

Filename

ChargingPoints.ttl ▾

Load Files Save File

Content editor: ChargingPoints.ttl

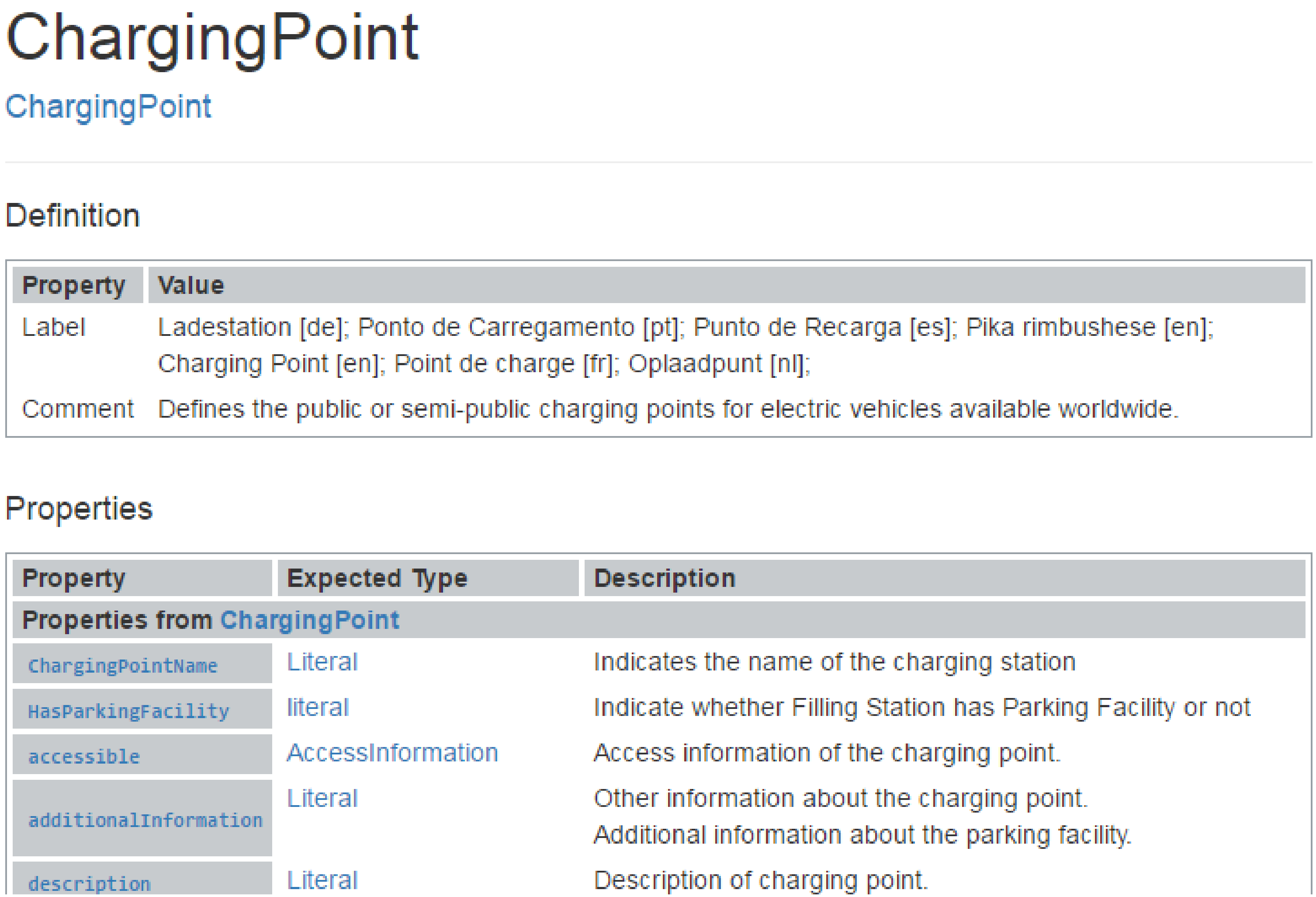
```
@prefix dct:     <http://purl.org/dc/terms/>.
@prefix foaf:    <http://xmlns.com/foaf/0.1/>.
@prefix s:       <http://schema.org/>.
@prefix gr:      <http://purl.org/goodrelations/v1#>.
@prefix vcard:   <http://www.w3.org/2001/vcard-rdf/3.0#> .
@prefix geo:     <http://www.w3.org/2003/01/geo/wgs84_pos#> .
@prefix skos:    <http://www.w3.org/2004/02/skos/core#> .

<http://purl.org/net/mobivoc/> a owl:Ontology .

# CLASSES

mv:AccessInformation
    a rdfs:Class, owl:Class ;
    rdfs:comment    "Access information of the charging point." ;
    rdfs:label "Access Information"@en, "Zugagsinformation"@de .

mv:AccessType
    a rdfs:Class, owl:Class ;
    rdfs:comment    "Describes access type of the charging point." ;
    rdfs:label "Access Type"@en .

mv:BookingType
    a rdfs:Class, owl:Class ;
```

● Syntax check passed.

Commit message

Turtle file edited, just for testing the editor.

VoCol: Documentation



ChargingPoint

ChargingPoint

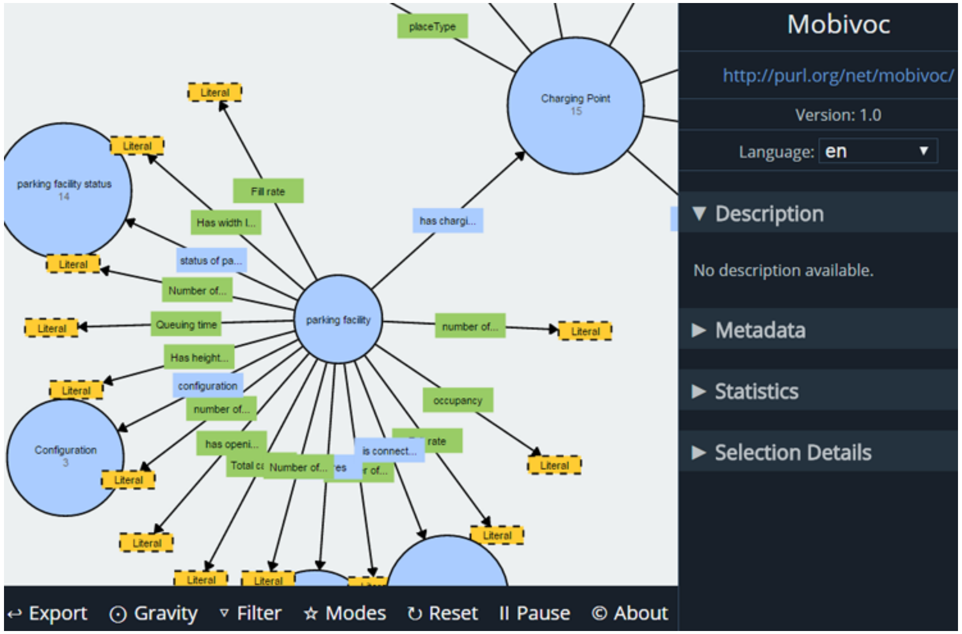
Definition

Property	Value
Label	Ladestation [de]; Ponto de Carregamento [pt]; Punto de Recarga [es]; Pika rimbushese [en]; Charging Point [en]; Point de charge [fr]; Oplaadpunt [nl];
Comment	Defines the public or semi-public charging points for electric vehicles available worldwide.

Properties

Property	Expected Type	Description
Properties from ChargingPoint		
ChargingPointName	Literal	Indicates the name of the charging station
HasParkingFacility	literal	Indicate whether Filling Station has Parking Facility or not
accessible	AccessInformation	Access information of the charging point.
additionalInformation	Literal	Other information about the charging point. Additional information about the parking facility.
description	Literal	Description of charging point.

VoCol: Visualization



Mobivoc

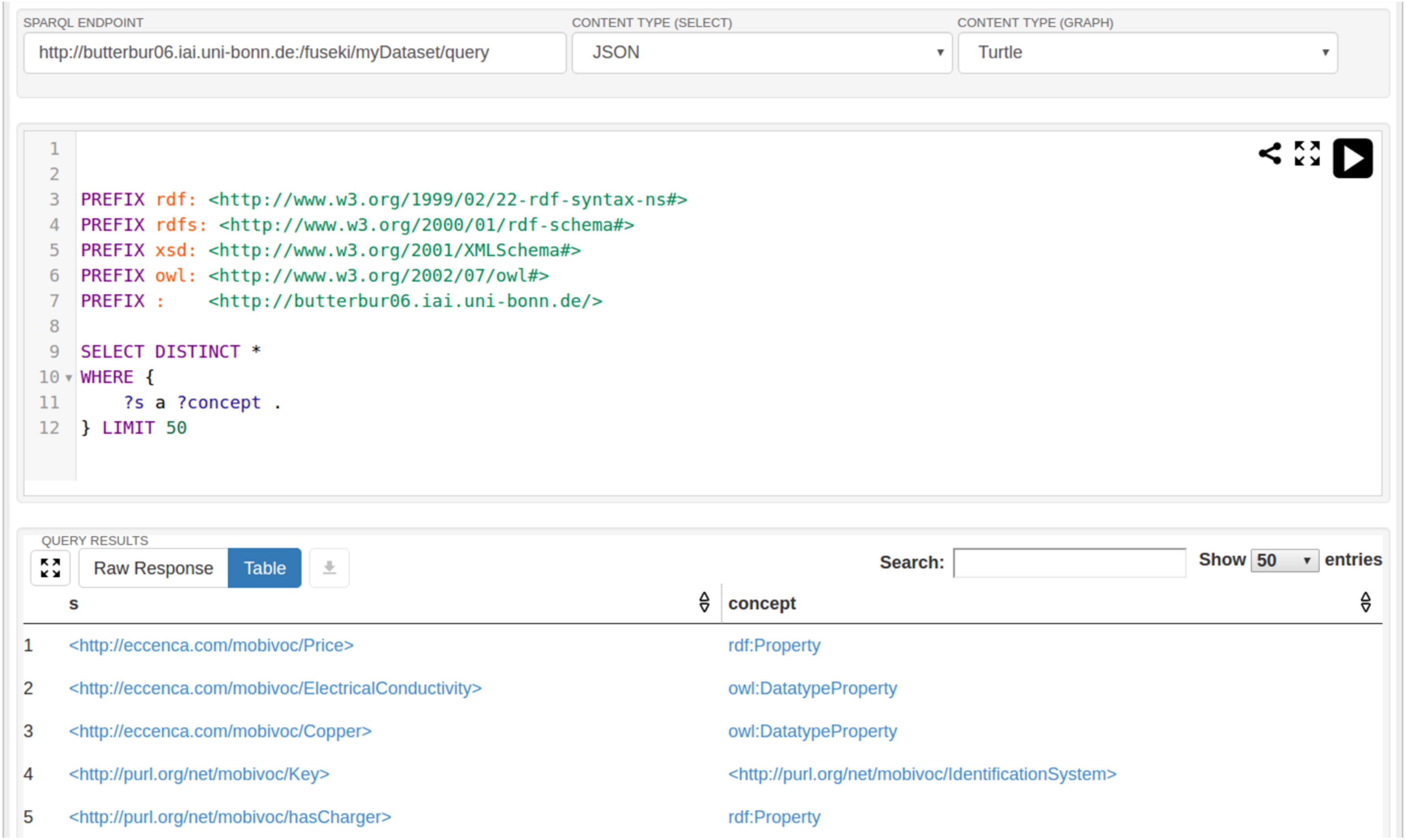
http://purl.org/net/mobivoc/

Version: 1.0

Language: en ▼

No description available.

VoCol: Querying



SPARQL ENDPOINT: http://butterbur06.iai.uni-bonn.de:/fuseki/myDataset/query

CONTENT TYPE (SELECT): JSON

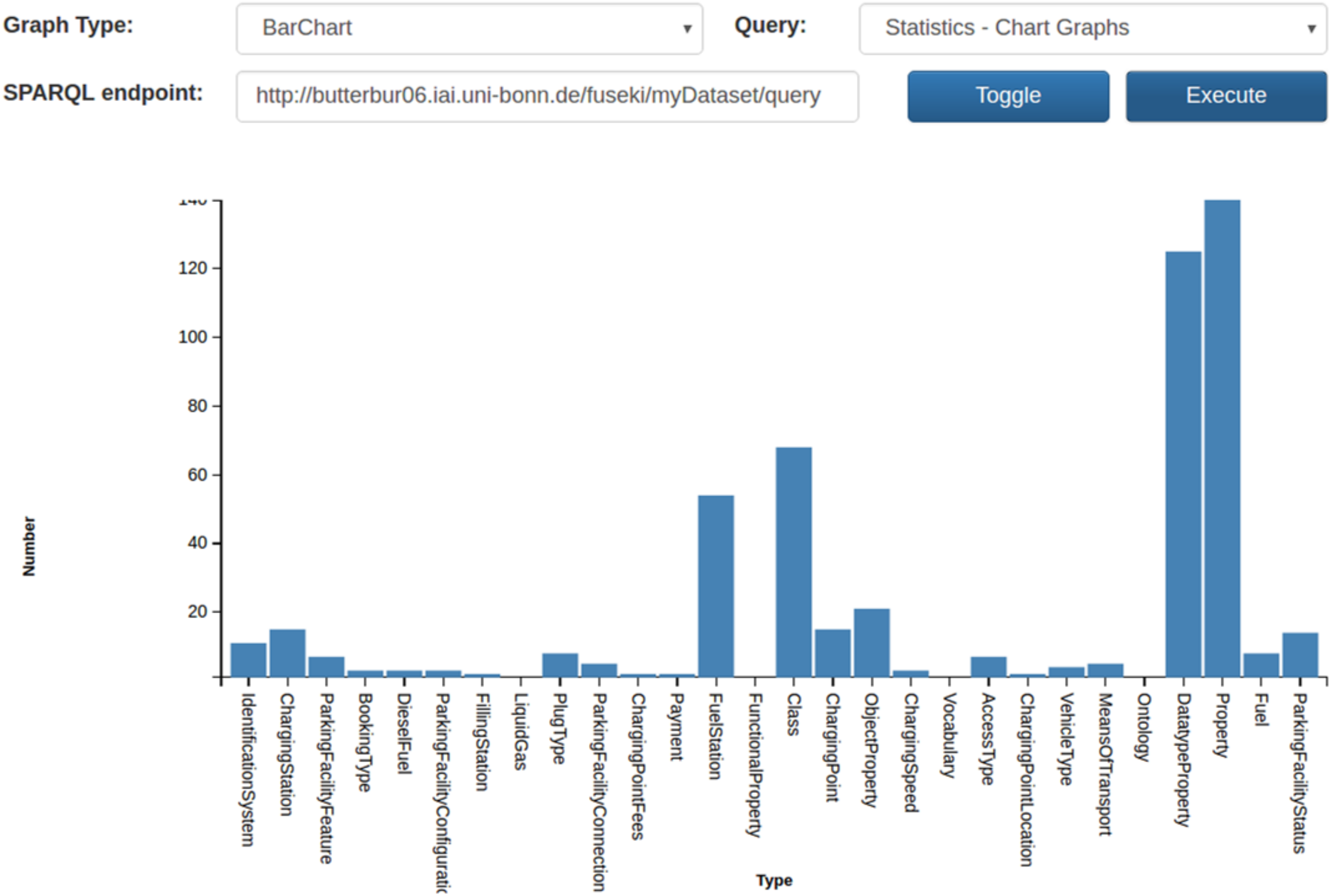
CONTENT TYPE (GRAPH): Turtle

```


PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX :    <http://butterbur06.iai.uni-bonn.de/>

SELECT DISTINCT *
WHERE {
    ?s a ?concept .
} LIMIT 50
```

QUERY RESULTS

	s	concept
1	<http://eccenca.com/mobivoc/Price>	rdf:Property
2	<http://eccenca.com/mobivoc/ElectricalConductivity>	owl:DatatypeProperty
3	<http://eccenca.com/mobivoc/Copper>	owl:DatatypeProperty
4	<http://purl.org/net/mobivoc/Key>	<http://purl.org/net/mobivoc/IdentificationSystem>
5	<http://purl.org/net/mobivoc/hasCharger>	rdf:Property

VoCol: Analytics

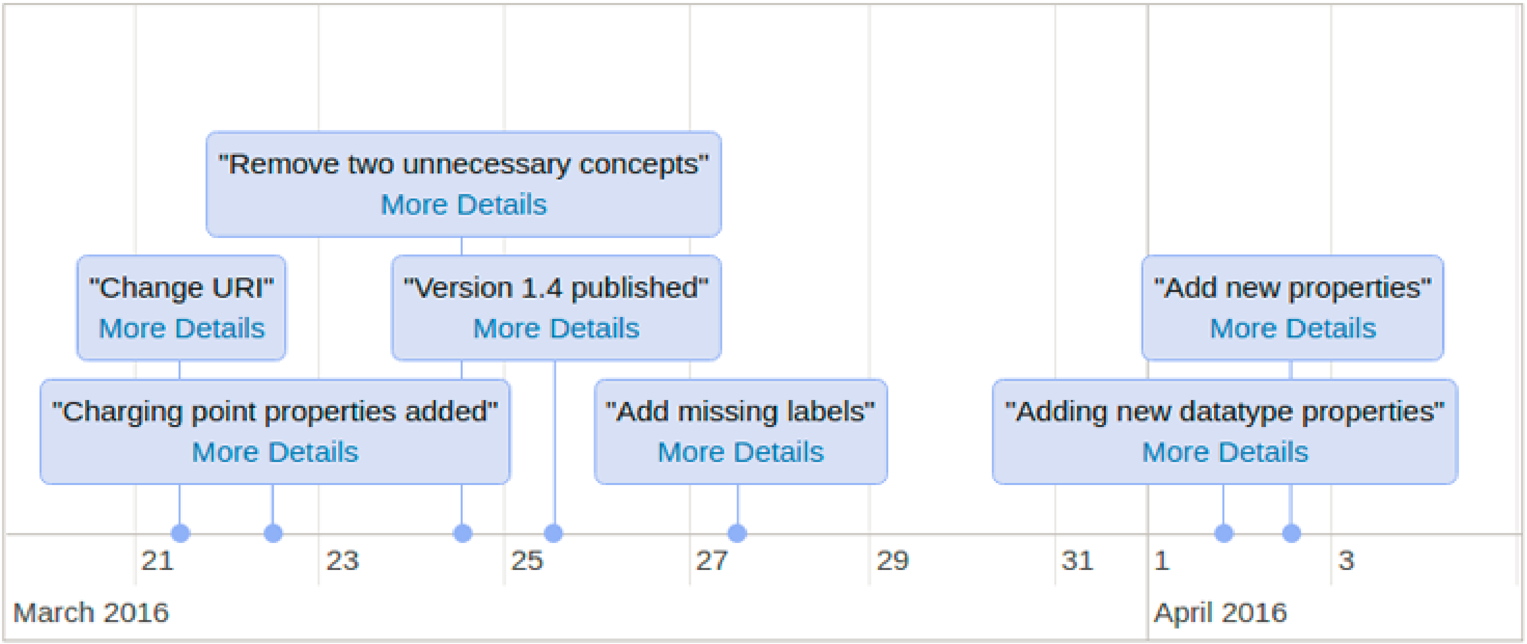


Graph Type: BarChart

Query: Statistics - Chart Graphs

SPARQL endpoint: http://butterbur06.iai.uni-bonn.de/fuseki/myDataset/query

VoCol: Evolution



Comment: Add new properties - Date: 02-04-2016

+ ObjectPropertyRange(<http://butterbur06.iai.uni-bonn.de/payment> <http://butterbur06.iai.uni-bonn.de/Payment>)

+ ObjectPropertyRange(<http://butterbur06.iai.uni-bonn.de/isLocated> <http://butterbur06.iai.uni-bonn.de/ParkingFacilityLocation>)

+ AnnotationAssertion(rdfs:label <http://butterbur06.iai.uni-bonn.de/SpecialParkingRestrictionsInForce> "besondere Parkbeschränkungen in Kraft"@de)

+ AnnotationAssertion(rdfs:label <http://butterbur06.iai.uni-bonn.de/isOwnedBy> "is owned by"@en)

+ DataPropertyRange(<http://butterbur06.iai.uni-bonn.de/threePhasedCurrentavailable> rdfs:Literal)

Application

- Industrial context (manufactoring company)
- Formally describe the assets of the company
- >10 people (knowledge engineers + domain experts)
- >80 issues, >250 classes, >400 properties, >180 instances
- R2RML mappings: vocabularies & legacy systems
- Queries against the legacy system, visualized results

+ different views on the vocabularies very helpful

Qualitative User Study

- 12 users – different levels of expertise
- Concurrent Think Aloud (CTA) method
- Tasks: define classes, properties, instances
- Commit changes locally, push in remote repository
- Test queries against SPARQL endpoint
- All VoCol functionalities were covered
- Post-study questionnaire
  (USE test, priorities, pros/cons, suggestions)

Qualitative User Study

+ „Very easy to learn and use“

+ „Very useful and effective support”

+ High usability (all USE scores > 4)

+ Turtle editor, syntax-checking and auto-completion

- More provenance information (author, date, etc.)

- Dynamic configuration

- Recommention of similar vocabularies

- ...

Conclusions

- Reuse existing VCSs as a core component of vocabulary development
- User-friendly client hiding the complexity of VCSs
- Comprehensive set of integrated services
- Loose coupling, webhooks, Vagrant & Docker
- Flexible architecture, easy to extend
- Further VCS, VoCol-as-service, etc.