

University of Oxford  
Department of Computer Science



# Connecting SMW to RDF Databases: Why, What, and How?



Markus Krötzsch  
University of Oxford

SMWCon 2011 Fall, Berlin



September 22, 2011



\*



\* Talk given during the 2011 papal visit to Berlin

# The “Semantic Web” Programme

- **Goal:** “Intelligent” automated processing of information on the Web without direct human intervention  
→ search, aggregation, question answering, ...

# The “Semantic Web” Programme

- **Goal:** “Intelligent” automated processing of information on the Web without direct human intervention  
→ search, aggregation, question answering, ...
- **Approach:**
  - Agree on Web-compatible standard data formats and protocols
  - Develop software tools for data gathering, storing, analysing, querying, ...
  - Establish best practices for data publishing and get a lot of data published

# Status as of 2011: The Web of Data

- **Established standards and best practices:**  
IRI (addressing), RDF(data), SPARQL (query),  
OWL (model), vocabularies and ontologies
- **Significant tool infrastructure:**  
Data stores/databases, query engines, inference  
engines/reasoners, libraries, crawlers, data browsers
- **Large data sources and consumers:**  
Sites publishing data feeds, search engines crawling  
for data, browsing/visualisation interfaces & mash-ups





# The Role of SMW

- SMW wikis as “miniature Semantic Webs”
- Data model similar to RDF
- Query model similar to OWL (queries = classes)
- Data feeds (RDF, JSON, RSS, ...)
- Limited mapping/integration features to other data collections

# But how does SMW fit into the Web of Data?



# RDF and SMW: Aligning Data Models

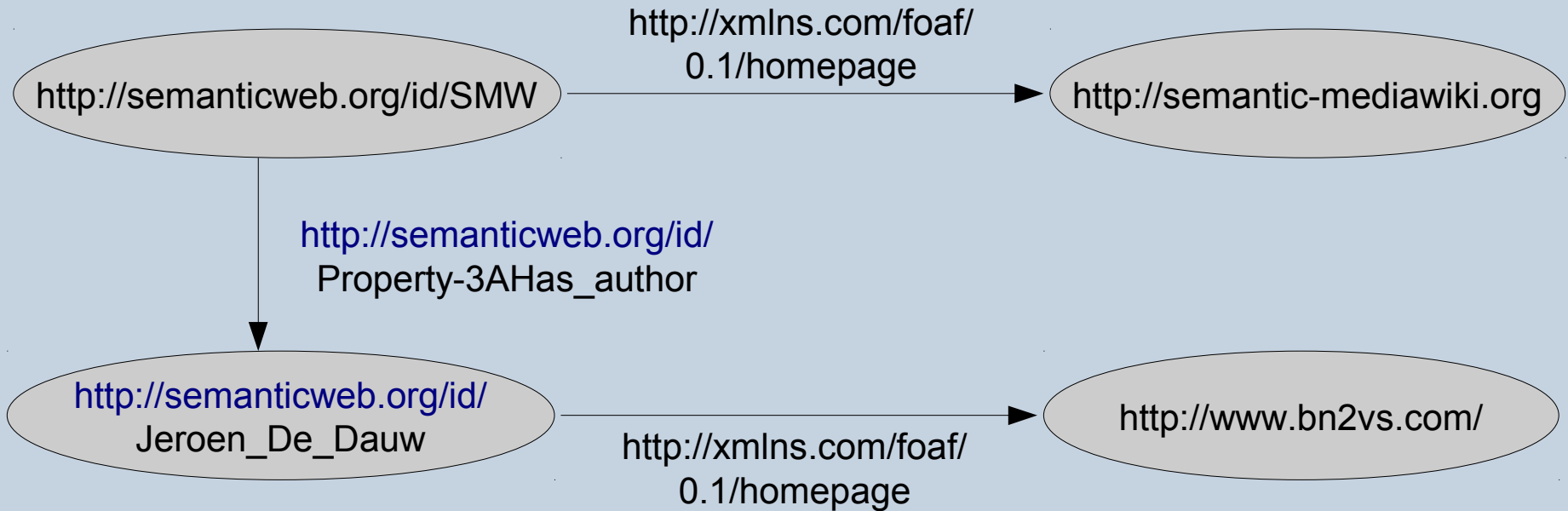
```

- <rdf:RDF>
- <owl:Ontology rdf:about="http://korrekt.org/page/Special:ExportRDF/Semantic_MediaWiki_%28Foundations_for_the_Web_of_Information_and_Services%29">
  <swivt:creationDate rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2011-09-22T09:49:29+02:00</swivt:creationDate>
  <owl:imports rdf:resource="http://semantic-mediawiki.org/swivt/1.0"/>
</owl:Ontology>
- <swivt:Subject rdf:about="http://korrekt.org/page/Special:URIResolver/Semantic_MediaWiki_(Foundations_for_the_Web_of_Information_and_Services)">
  - <rdfs:label>
    Semantic MediaWiki (Foundations for the Web of Information and Services)
  </rdfs:label>
  <swivt:page rdf:resource="http://korrekt.org/page/Semantic_MediaWiki_(Foundations_for_the_Web_of_Information_and_Services)"/>
  <rdfs:isDefinedBy rdf:resource="http://korrekt.org/page/Special:ExportRDF/Semantic_MediaWiki_(Foundations_for_the_Web_of_Information_and_Services)"/>
  <rdf:type rdf:resource="http://korrekt.org/page/Special:URIResolver/Category-3APublication"/>
  <swivt:wikiNamespace rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">0</swivt:wikiNamespace>
  <property:Author1 rdf:resource="http://korrekt.org/page/Special:URIResolver/Markus_Kr-C3-B6tzsch"/>
  <property:Author2 rdf:resource="http://korrekt.org/page/Special:URIResolver/Denny_Vrandecic"/>
- <property:Booktitle rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
  Foundations for the Web of Information and Services
</property:Booktitle>
<property:Date rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2011-01-01T00:00:00</property:Date>
<property:Editor1 rdf:resource="http://korrekt.org/page/Special:URIResolver/Dieter_Fensel"/>
<property>Last_update rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2011-08-02T00:00:00</property>Last_update>
<swivt:wikiPageModificationDate rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2011-08-02T13:04:29</swivt:wikiPageModificationDate>
<property:Publication_type rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Book chapter</property:Publication_type>
<property:Published_at rdf:resource="http://korrekt.org/page/Special:URIResolver/Foundations_for_the_Web_of_Information_and_Services"/>
<property:Publisher rdf:resource="http://korrekt.org/page/Special:URIResolver/Springer"/>
<property:Title rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Semantic MediaWiki</property:Title>
<property:Topic rdf:resource="http://korrekt.org/page/Special:URIResolver/Semantic_wikis"/>
</swivt:Subject>

```

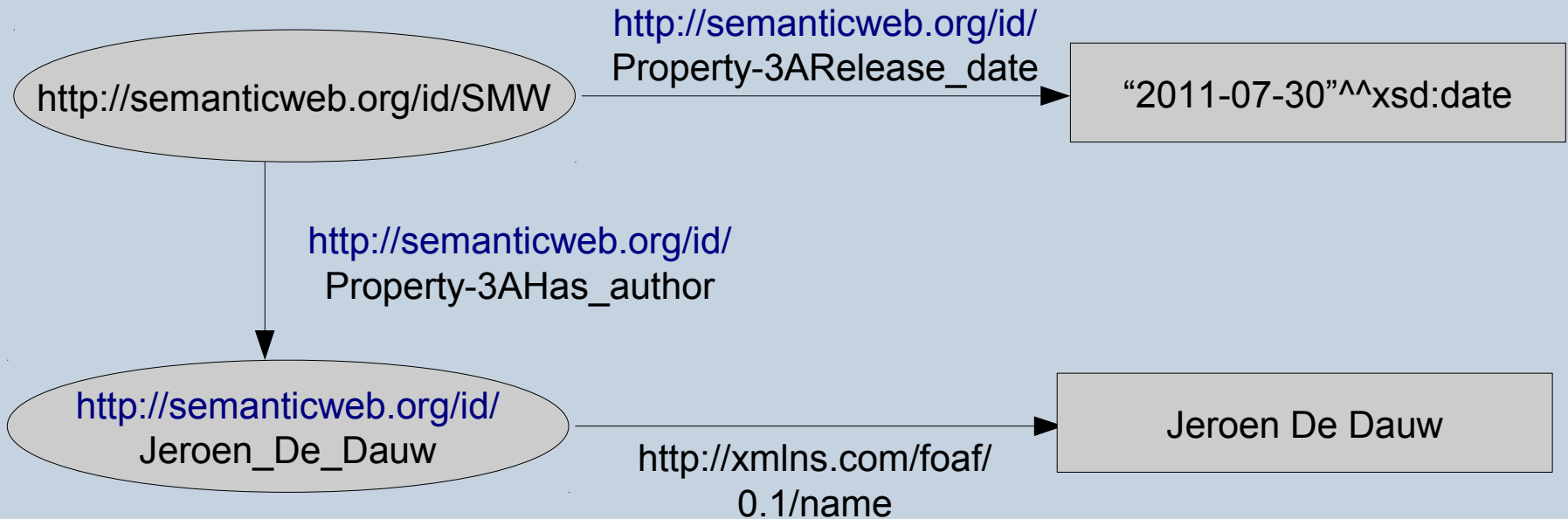
# RDF Data Model

- RDF: data as directed graph, labelled with IRIs:



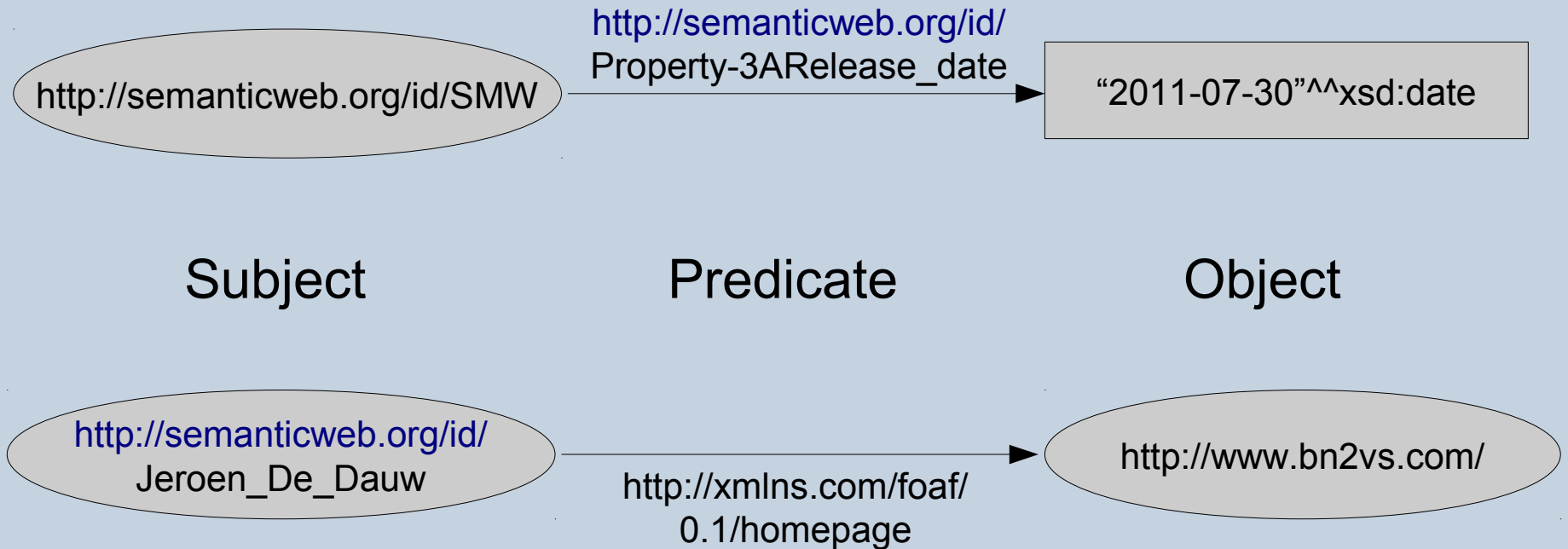
# RDF Data Model

- RDF: datatypes and literals



# RDF Data Model

- RDF: graphs consist of triples:



# Data model: RDF vs. SMW

- Subject-Predicate-Object  $\approx$  Page-Property-Value
  - Difference: SMW properties have fixed datatypes
  - Structure of data more constrained in SMW
- IRIs and literals  $\approx$  SMW dataitems
  - Similarity: subjects must be IRIs/wiki pages
- RDF datatypes  $\neq$  SMW datatypes
  - Different, incomparable type set
  - Many-to-many relationship

# Other Data Models

- SQL
  - Relational model: data in tables
  - Strong emphasis on schema
  - No real data exchange/publication/syndication
- JSON
  - Object model
  - Very flexible, schema-less, little datatype support
  - Exchange syntax, no standard for object identifiers

# RDF and SMW: Data Access and Querying

# Getting Access to Data

- Main task of SMW: query answering
- Internal query language: ASK
- Different data models provide different access paths:
  - Relational model: SQL
  - RDF: SPARQL
  - JSON: no standard; some noSQL approaches

# Layers of an ASK Query

```
{ { #ask: [[Category:City]] [[Population::>3000000]]  
  | ?Population  
  | ?Location of  
  | format=broadtable  
  | sort=Population  
}
```

- 1) Which pages are query results?
- 2) What other information should be included?
- 3) How should the result be presented?

# Layers of an ASK Query

- 1) Which pages are query results?
  - 2) What other information should be included?
  - 3) How should the result be presented?
- SQL and SPARQL cover (1)
  - Presentation (3) is achieved by higher level libraries, e.g. SPARK for SPARQL
  - (2) not directly supported

# SMW vs. SPARQL/SQL Result Tables

```
{{ #ask: [[Category:City]] [[Population::>3000000]]  
| ?Population | ?Twin city | ?Location of  
}}
```

## Result in SMW: 3-dimensional table

	Population	Twin city	Location of
Berlin	3471756	Paris, London, Tokio, Buenos Aires, ...	SMWCon, Berlin Marathon, ...
...	...	...	...

# SMW vs. SPARQL/SQL Result Tables

```
{{ #ask: [[Category:City]] [[Population::>3000000]]  
| ?Population | ?Twin city | ?Location of  
}}
```

**Result in SQL/SPARQL:** 2-dimensional table

	Population	Twin city	Location of
Berlin	3471756	Paris	SMWCon
Berlin	3471756	London	SMWCon
...	...	...	...
Berlin	3471756	Paris	Berlin Marathon
...	...	...	...

# Comparing Page Selection Queries (1)

- [[Category:City]] [[Population::>3000000]]
- SQL

```
SELECT DISTINCT t5.smw_title AS t,t5.smw_namespace AS ns
FROM `smw_ids` AS t5
      INNER JOIN `smw_inst2` AS t1 ON t5.smw_id=t1.s_id
      INNER JOIN `smw_atts2` AS t3 ON t1.s_id=t3.s_id
WHERE ( t1.o_id='738' AND
        ((t3.value_num>='3000000') AND t3.p_id='363') );
```

# Comparing Page Selection Queries (1)

- `[[Category:City]] [[Population::>3000000]]`
- SPARQL

```
SELECT DISTINCT ?result
WHERE {
    ?result rdf:type wiki:Category-3ACity .
    ?result property:Population ?v1 .
    FILTER( ?v1 >= "3000000"^^xsd:double )
}
```

# Comparing Page Selection Queries (1)

- `[[Category:City]] [[Population::>3000000]]`
- OWL

```
ObjectIntersectionOf(  
  wiki:Category-3ACity  
  DataSomeValuesFrom( property:Population  
    DatatypeRestriction( xsd:int  
      xsd:minInclusive "3000000"^^xsd:int )  
  )  
)
```

# Possible Query Languages for SMW

- SQL
  - Many stable, large-scale implementations
  - Very different from ASK
  - Resulting SQL queries are unusual for RDBMS
- SPARQL
  - Numerous *mostly* stable, *reasonably* large-scale implementations
  - Quite similar to ASK
  - Many SMW queries quite typical for SPARQL
- OWL
  - Numerous stable, highly-optimized implementations
  - Most similar to ASK, but limited on FILTER features
  - Implementations not optimised for big, changing data

# SMW & SPARQL: An *almost* perfect match

- Not all SMW-data is naturally triples
  - Example: page namespaces not stored like this
- Different capabilities of SQL and SPARQL
  - Examples: Random order, pattern matching, geographic coordinates support
- Data formats in SQL and SPARQL are not the same
  - Example: Type:String and Type:Text
- Aggregation (count) in SPARQL tools still shaky

# SMW & SPARQL: An *almost* perfect match

- Some query structures hard to express in SPARQL:

```
{{#ask: [[population::>80,000,000]] OR [[Paris]] }}
```

# SMW & SPARQL: An *almost* perfect match

- Some query structures hard to express in SPARQL:

```
{{#ask: [[population::>80,000,000]] OR [[Paris]] }}
```

```
SELECT DISTINCT ?result WHERE
```

```
{ ?result swivt:page ?url .
```

```
  OPTIONAL
```

```
    { ?v2 property:Population ?v1 .
```

```
      FILTER( ?v1 >= "80000000"^^xsd:double ) }
```

```
  FILTER( ?result = wiki:Paris || ?result = ?v2 )
```

```
}
```



# RDF and SMW: Support in SMW 1.6.0

# Using RDF Databases with SMW

- RDF data model very close to SMW
  - SPARQL query model close to ASK
- Use RDF/SPARQL database to manage SMW data
- Supported by new SPARQL 1.1 Update
  - Available since SMW 1.6.0
  - Benefits:
    - Better query performance (hopefully)
    - Additional features (e.g. SPARQL endpoint)
    - More flexible integration with other local data stores

# Popular RDF Databases (September 2011)

- Various advanced platforms to choose from:
  - 4Store (free OSS)
  - 5Store (commercial)
  - AllegroGraph (commercial)
  - OWLIM (commercial, some free downloads)
  - Virtuoso (free OSS, commercial support)
  - Oracle (commercial)
  - Jena
  - Sesame
- SMW integration first for 4Store
  - Should work with most SPARQL 1.1 stores, maybe adjustments needed, feedback welcome

# Installation and Configuration

## (1) Install RDF store

# Installation and Configuration

- (1) Install RDF store
- (2) Configure store to run in server mode, offering SPARQL services for read and write

# Installation and Configuration

- (1) Install RDF store
- (2) Configure store to run in server mode, offering SPARQL services for read and write
- (3) Configure SMW to use these services by giving URLs:

```
$smwgDefaultStore = 'SMWSparqlStore';  
$smwgSparqlDatabase = 'SMWSparqlDatabase4Store'; // for 4Store  
$smwgSparqlQueryEndpoint = 'http://localhost:8080/sparql/';  
$smwgSparqlUpdateEndpoint = 'http://localhost:8080/update/';  
$smwgSparqlDataEndpoint = 'http://localhost:8080/data/';
```

# Current Limitations

- RDF store only for #ask page selection
  - Basic lookups still done in SQL
  - All semantic data mirrored in SQL and RDF
- No support for Concepts
- No SMW emulation for inferencing as in SQL (subclass, subproperty); though the RDF store might have it
  - Equality (redirect) resolution always supported

# RDF and SMW: Possible Futures

# SPARQL in and SPARQL out

- SPARQL service support
  - Native (hard) or brokered from RDF store (easy)
  - Currently available directly from RDF store only
- Inline SPARQL
  - Use SPARQL queries like `#ask` on pages
  - Pull in data from external sources
  - Formatting like in SMW
  - Partial solutions: SPARQL SMW extensions, Spark

# Wikitext-independent Data

- Allow data that does not come from wiki pages to co-exist in the store
- Visible in queries
- Possibly editable via interface

SMW as “Web frontend to a semantic data store”?

# Mapping

- Data integration across data sources unsolved
- Current vocabulary import does not work with RDF store support (no inverse lookup)
  - How should resources be mapped?
- Dynamic discovery of related sources?
  - Prototype project: Shortipedia



# Connecting SMW to RDF Databases: Why, What and How?

# Connecting SMW to RDF Databases: Why, What, How ... and Who?

