Chapter 4
Creating Linked Data from Relational Databases

NIKOLAOS KONSTANTINOU

DIMITRIOS-EMMANUEL SPANOS

Outline

Introduction (1)

Relational databases vs. Semantic Web standards

- Active research topic since more than a decade ago
- Not just a theoretical exercise, but also practical value
  - Bootstrap the Semantic Web with a sufficiently large mass of data
  - Facilitate database integration
  - Ontology-based data access
  - Semantic annotation of dynamic Web pages

Introduction (2)

Database-to-ontology mapping

- The investigation of the similarities and differences among relational databases and Semantic Web knowledge models
- Broad term encompassing several distinct problems
- Classification of approaches needed

Outline

Semantic Annotation of Dynamic Web Pages (1)

Goal of the Semantic Web: emergence of a Web of Data, from the current Web of Documents

HTML documents are mainly for human consumption

How to achieve this?

- Add *semantic* information to HTML documents
  - I.e. setup correspondences with terms from ontologies

RDFa

- Embedding references to ontology terms in XHTML tags, *but*…

Semantic Annotation of Dynamic Web Pages (2)

What about dynamic documents?

- Content retrieved from relational databases
- The biggest part of the World Wide Web
  - Aka. Deep Web
- CMS's, forums, wikis, etc.
- Manual annotation of every single dynamic web page is infeasible

Semantic Annotation of Dynamic Web Pages (3)

Directly “annotate” the database schema!

- Establish correspondences between the elements of the database schema and a suitable existing domain ontology

Use these correspondences to generate automatically semantically annotated dynamic pages

Heterogeneous Database Integration (1)

Longstanding issue in database research

- Due to differences in:
  - Software infrastructure
  - Syntax
  - Representation models
  - Interpretation of the same data

Remains unresolved to a large degree

Heterogeneous Database Integration (2)

Typical database integration architecture

- One or more conceptual models for the description of the contents of each source database
- Queries against a global conceptual schema
- Wrappers on top of every source database for the reformulation of queries and data retrieval

Heterogeneous Database Integration (3)

Ontology-based database integration

- Ontologies instead of conceptual schemas
- Definition of correspondences between source databases and one or more ontologies
- LAV, GAV or GLAV approach (target schema = ontology)
  - Database term ↔ Query over the ontology (LAV)
  - Ontology term ↔ Query over the database (GAV)
  - Query over the database ↔ Query over the ontology (GLAV)
- Mappings between relational database schemas and ontologies need to be discovered!

Ontology-Based Data Access (1)

Objective:

- Offer high-level services on top of an information system without knowledge of the underlying database schema

Ontology as an intermediate layer between the end user and the storage layer

- Ontology provides an abstraction of the database contents
- Users formulate queries using terms from the ontology

Ontology-Based Data Access (2)

Similar to a database integration architecture

- OBDA engine ≈ wrapper
    - Transforms queries against the ontology to queries against the local data source

OBDA engine

- Performs query rewriting
- Uses mappings between a database and a relevant domain ontology

Advantages

- Semantic queries posed directly to the database
- No need to replicate database contents in RDF

Semantic Rewriting of SQL Queries

Objective:

- Reformulate an SQL query to another one that better captures the intention of the user

Substitution of terms in the original SQL query with synonyms and related terms from an ontology

Also related:

- Query relational data using external ontologies as context
  - SQL queries with their WHERE conditions containing terms from an ontology

Feature implemented in some DBMSes

- E.g. OpenLink Virtuoso, Oracle

Mass Data Generation for the Semantic Web

Reasons for slow uptake of the Semantic Web

- Few successful paradigms of tools and “killer” applications
- Few data
- “Chicken-and-egg” problem

Relational databases hold the majority of data on the World Wide Web

Automated extraction of RDB contents in RDF

Generation of a critical mass of Semantic Web data

Increased production of SW applications and tools anticipated

Ontology Learning (1)

Manual development of ontologies is difficult, time-consuming and error-prone

Ontology learning

- Semi-automatic extraction of ontologies from free texts, semi-structured documents, controlled vocabularies, thesauri etc.
- Relational databases can be sources of domain knowledge as well
- Information gathered from database schema, contents, queries and stored procedures
- Supervision from domain expert is necessary

Ontology Learning (2)

Useful in domains where there is no suitable ontology

- Typical in the earlier Semantic Web years

Nowadays, ontology learning for the creation of a “wrapping” ontology for an RDB in:

- OBDA
- Database integration

Intended Meaning of a Relational Schema (1)

Database schema design

- Conceptual model ⟶ relational model
- Subsequent changes often directly to the relational model
- Initial conceptual model lost
- Hard to re-engineer to another model (e.g. object-oriented)

Definition of correspondences between RDB and ontology

- Semantic grounding of the meaning of the former

Intended Meaning of a Relational Schema (2)

Facilitates:

- Database maintenance
- Integration with other data sources
- Mapping discovery between 2 or more database schemas

In the latter case, database-to-ontology mappings are used as a reference point for the construction of inter-database schema mappings

Database Integration with Other Data Sources

Mapping RDB to RDF enables integration with existing RDF content

- Content generated from either structured or unstructured sources

Linked Data paradigm

- Vocabulary reuse
- Inter-dataset links
- Identifier reuse
- Facilitates data source integration at global level
- Billions of RDF statements from several domains of interest

Integration of RDB content with Linked Data offers unlimited potential

Outline

Existing Classifications (1)

Several classification schemes proposed for database-to-ontology mapping approaches

Classification criteria vs. descriptive measures

- Classification criteria
  - Finite number of values
  - Should separate approaches in non-overlapping sets
- Descriptive measures
  - Can also be qualitative

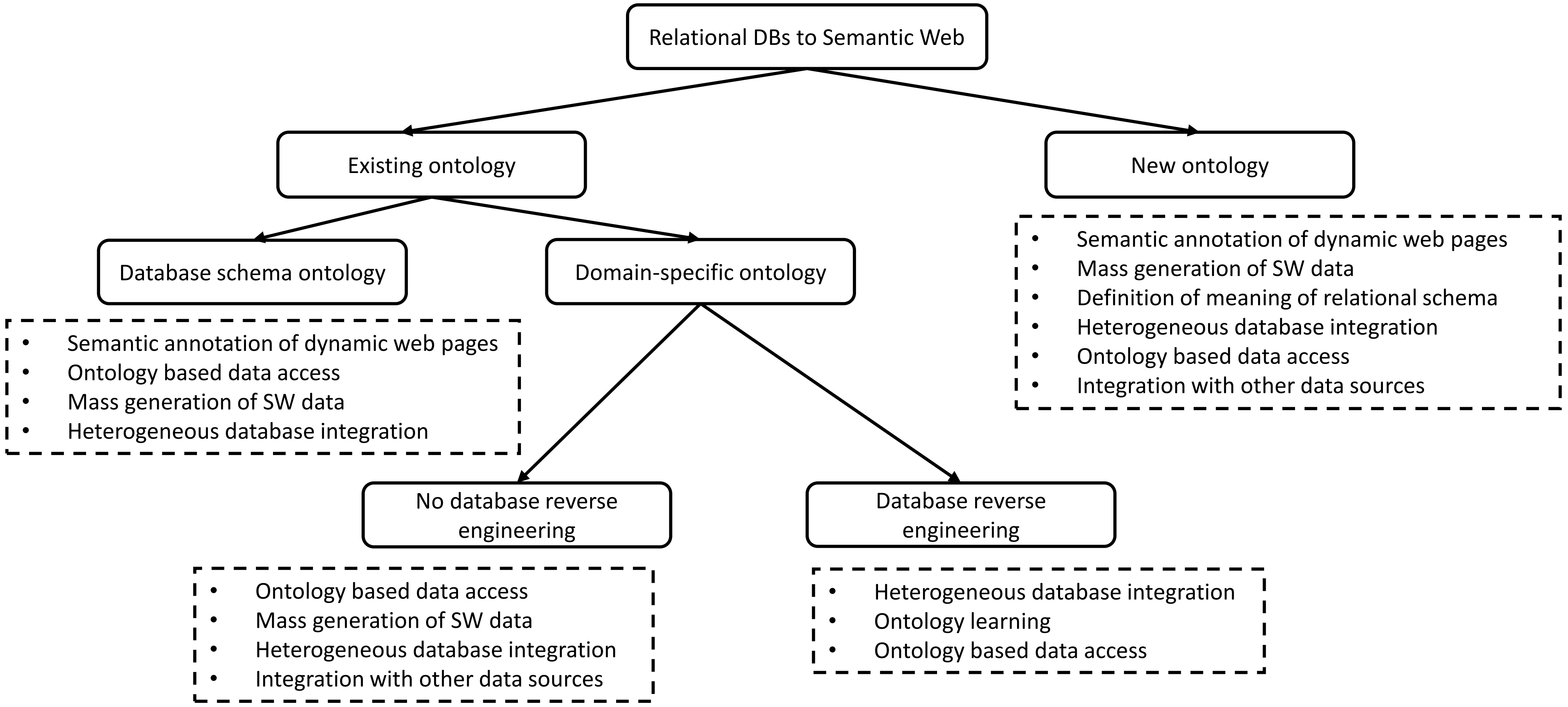
Existing Classifications (2)

Work	Classification criteria	Values	Descriptive parameters
(Auer et al. 2009)	a. Automation in the creation of mapping b. Source of semantics considered c. Access paradigm d. Domain reliance	a. Automatic/Semi-automatic/Manual b. Existing domain ontologies/Database/Database and User c. Extract-Transform-Load (ETL)/SPARQL/Linked Data d. General/Dependent	Mapping representation language
(Barrasa-Rodriguez and Gómez-Pérez 2006)	a. Existence of ontology b. Architecture c. Mapping exploitation	Yes (ontology reuse)/No (created ad-hoc) Wrapper/Generic engine and declarative definition Massive upgrade (batch)/Query driven (on demand)	-
(Ghawi and Cullot 2007)	a. Existence of ontology b. Complexity of mapping definition c. Ontology population process d. Automation in the creation of mapping	a. Yes/No b. Complex/Direct c. Massive dump/Query driven d. Automatic/Semi-automatic/Manual	Automation in the instance export process
(Hellmann et al. 2011)	-	-	Data source, Data exposition, Data synchronization, Mapping language, Vocabulary reuse, Mapping automation, Requirement of domain ontology, Existence of GUI

Existing Classifications (3)

Work	Classification criteria	Values	Descriptive parameters
(Konstantinou et al. 2008)	a. Existence of ontology b. Automation in the creation of mapping c. Ontology development	a. Yes/No b. Automatic/Semi-automatic/Manual c. Structure driven/Semantics driven	Ontology language, RDBMS supported, Semantic query language, Database components mapped, Availability of consistency checks, User interaction
(Sahoo et al. 2009)	Same as in (Auer et al. 2009) with the addition of:		
	a. Query implementation b. Data integration	a. SPARQL/SPARQL→SQL b. Yes/No	Mapping accessibility, Application domain
(Sequeda et al. 2009)	-	-	Correlation of primary and foreign keys, OWL and RDFS elements mapped
(Zhao and Chang 2007)	a. Database schema analysis	Yes/No	Purpose, Input, Output, Correlation analysis of database schema elements, Consideration of database instance, application source code and other sources

A Proposed Classification (1)


Relational DBs to Semantic Web
Existing ontology
New ontology
Database schema ontology
Domain-specific ontology
No database reverse engineering
Database reverse engineering

Database schema ontology:

- Semantic annotation of dynamic web pages
- Ontology based data access
- Mass generation of SW data
- Heterogeneous database integration

New ontology:

- Semantic annotation of dynamic web pages
- Mass generation of SW data
- Definition of meaning of relational schema
- Heterogeneous database integration
- Ontology based data access
- Integration with other data sources

No database reverse engineering:

- Ontology based data access
- Mass generation of SW data
- Heterogeneous database integration
- Integration with other data sources

Database reverse engineering:

- Heterogeneous database integration
- Ontology learning
- Ontology based data access

A Proposed Classification (2)

Total classification of all relevant solutions in mutually disjoint classes

Exceptions

- Customizable software tools with multiple possible workflows
- Each one belongs to multiple categories

Every class associated with a number of benefits/motivations

- Not significant correlation among taxonomy classes and motivations and benefits
- Categorization of approaches based on the *nature* of the mapping and the *techniques applied* to establish the mapping
- Benefits state the *applications* of the already established mappings

Classification Criteria (1)

Existence of ontology

- Is an ontology *required* for the application of the approach?
- Yes
  - Establishment of mappings between a given relational database and a given existing ontology
  - Domain of ontology compatible with database domain
  - Existing ontology selected by human user
- No
  - Creation of a new ontology from a given relational database
  - Useful when:
    - An ontology for the domain covered by the database is not available yet
    - The human user is not familiar with the domain of the database and relies on the mapping process to discover the semantics of the database contents

Classification Criteria (2)

Domain of the generated ontology

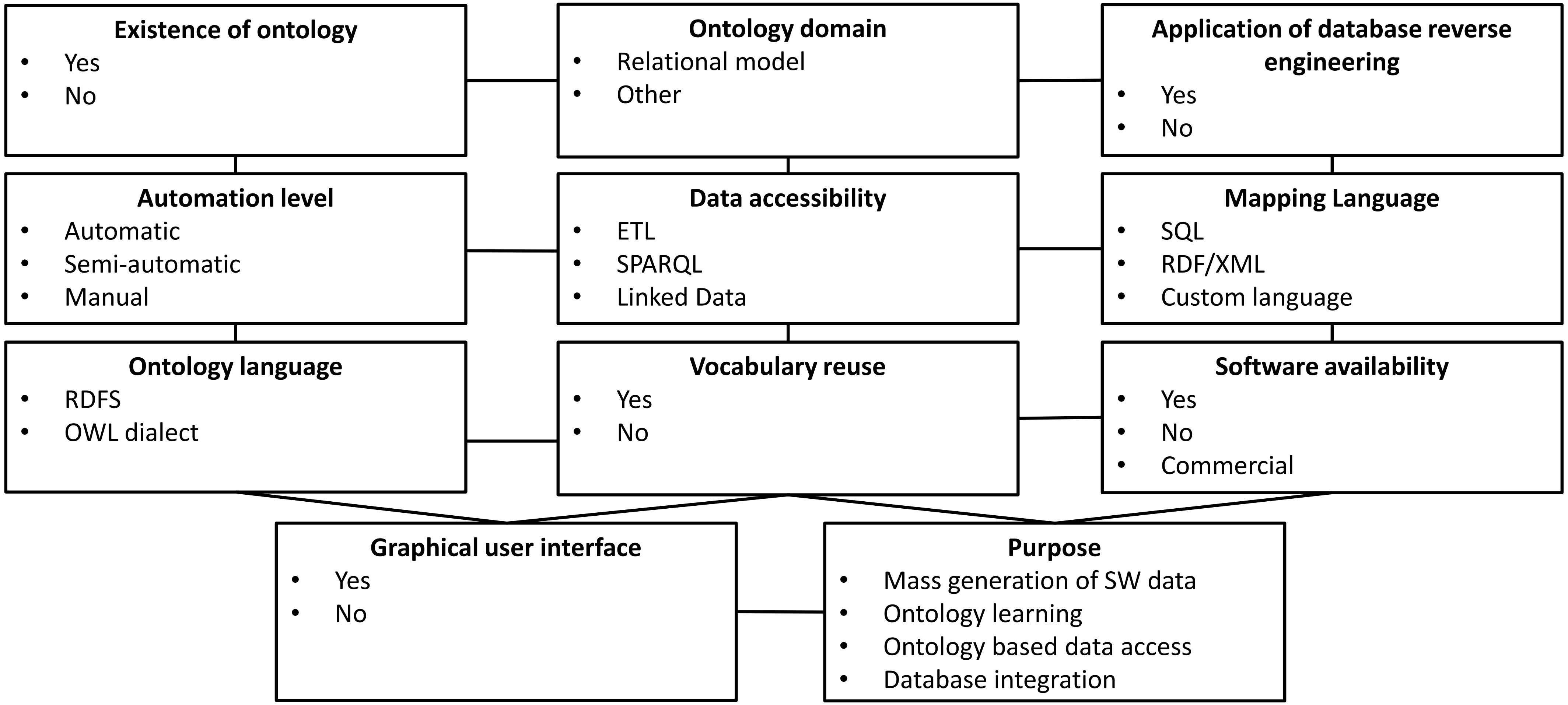
- What is the domain of the generated ontology?
- The relational model
  - Generated ontology consists of concepts and relationships that reflect the constructs of the relational model
  - Mirrors the structure of the input relational database
  - “*Database schema ontology*”
  - Mainly automatic class of approaches
- Another domain
  - Depending on the domain described by the contents of the input database

Classification Criteria (3)

Database reverse engineering

- Are any database reverse engineering techniques applied?
- Yes
  - Recover the initial conceptual schema from the relational schema
  - Translate re-engineered schema to an ontology expressed in a target language
- No
  - Few basic translation rules from the relational to the RDF model
  - Reliance on the human expert for the definition of complex mappings and the enrichment of the generated ontology

Classification criteria and descriptive features



Existence of ontology
- Yes
- No

Ontology domain
- Relational model
- Other

Application of database reverse engineering
- Yes
- No

Automation level
- Automatic
- Semi-automatic
- Manual

Data accessibility
- ETL
- SPARQL
- Linked Data

Mapping Language
- SQL
- RDF/XML
- Custom language

Ontology language
- RDFS
- OWL dialect

Vocabulary reuse
- Yes
- No

Software availability
- Yes
- No
- Commercial

Graphical user interface
- Yes
- No

Purpose
- Mass generation of SW data
- Ontology learning
- Ontology based data access
- Database integration

Descriptive Features (1)

Level of Automation

- How much is the user involved in the mapping process?
- Automatic
  - No input from human user
- Semi-automatic
  - Some input from human user
  - Sometimes necessary
  - Sometimes optional (e.g. validation or enrichment of results)
- Manual
  - Mapping defined entirely from human user
- Feature usually common among approaches of the same class

Descriptive Features (2)

Data Accessibility

- The way the mapping result is accessed
  - Aka. access paradigm / mapping implementation / data exposition
- ETL
  - Result of the mapping process generated and stored as a whole in an external storage medium (i.e. *materialized*)
  - Aka. batch transformation / massive dump

Descriptive Features (3)

Data Accessibility (cont'd)

- SPARQL
  - Only a part of the mapping result is accessed
  - No additional storage medium is required (i.e. no materialization)
  - Rewriting of a SPARQL query to an SQL one
  - SQL results transformed back to SPARQL results
  - Aka. query-driven access
- Linked Data
  - Mapping result published as Linked Data (i.e. all URIs use the HTTP scheme and, when dereferenced, provide useful information for the resource they identify)

Descriptive Features (4)

Data Synchronization

- Does the mapping result reflect the current database contents?
- Static
  - Mapping executed only once
  - Mapping result not tied with source database
- Dynamic
  - Mapping executed on every incoming query
  - Mapping result depends on current database state
- Strongly related to data accessibility, redundant feature
- ETL methods are static
- SPARQL (query-driven) and Linked Data methods are dynamic

Descriptive Features (5)

Mapping language

- The language in which the mapping is represented
- Large variance of values: a lot of proprietary formats
- …until the standardization of R2RML
- Feature only applicable to methods that need to reuse the mapping
    - E.g. not applicable to ontology generation methods

Descriptive Features (6)

Ontology language

- The language in which the involved ontology is expressed
- Either:
  - The language of the ontology generated by the approach
  - The language of the existing ontology required
- RDFS
- OWL (all flavours and dialects)

Descriptive Features (7)

Vocabulary reuse

- Does the mapping support more than one existing ontologies?
- Yes
  - Mainly manual approaches
  - Human user free to reuse terms from existing ontologies
  - Not obligatory to reuse terms
- No
  - E.g. methods generating a new “database schema ontology”

Descriptive Features (8)

Software availability

- Does the method have a free implementation?
- Theoretical methods
- Practical solutions
- Commercial software

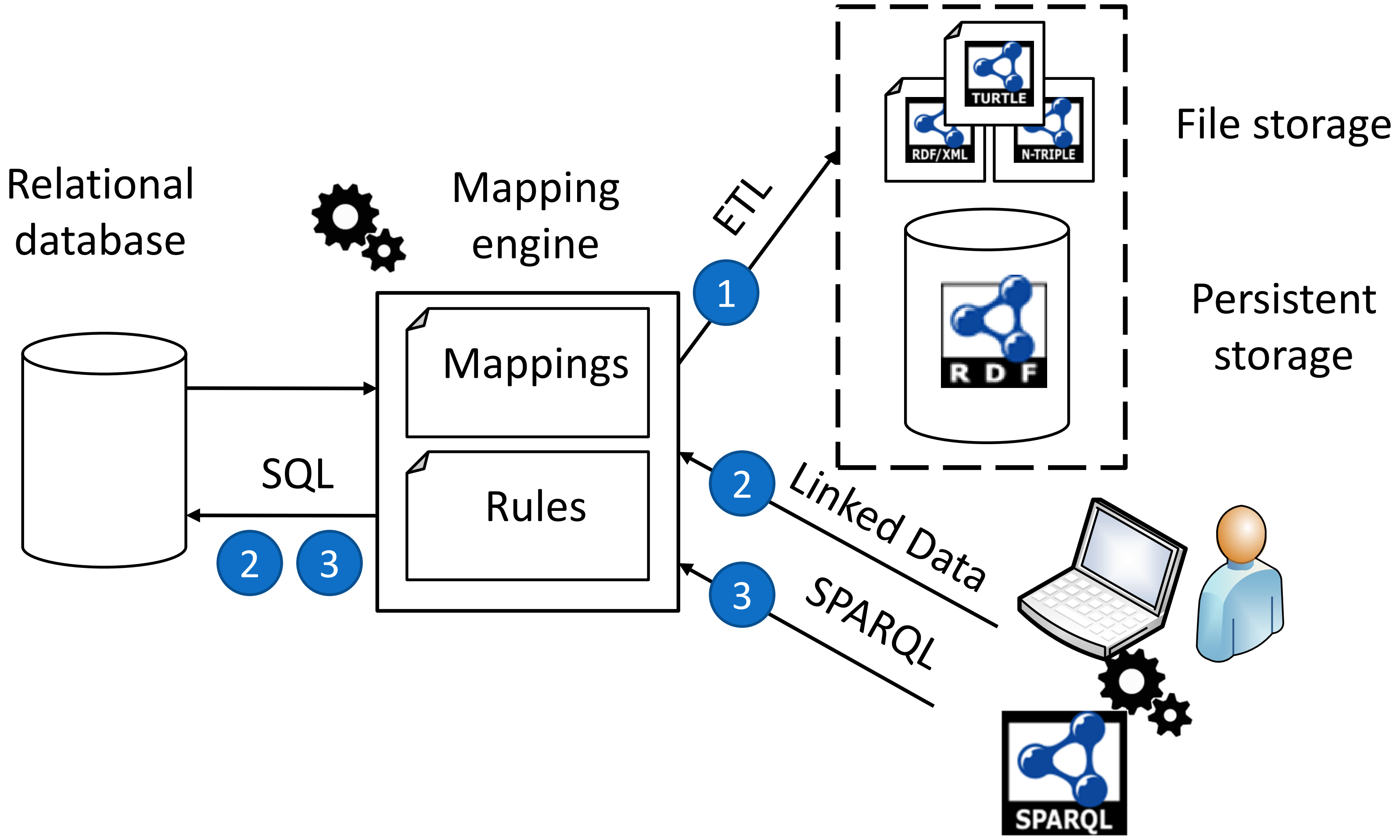
Descriptive Features (9)

Graphical User Interface

- Can the user interact with the system via a GUI?
- Feature applicable to approaches with an accessible software implementation
- Guides user through steps of the mapping process
- Provides mapping suggestions
- Essential for inexperienced users / users not familiar with SW technologies

Outline

Creating Ontology and Triples from a Relational Database (1)



Creating Ontology and Triples from a Relational Database (2)

Generation of a new ontology

Population with RDF data originating from the database

Mapping engine

- Communicates with database
- Uses heuristic or manually defined rules

3 ways to access the generated RDF data

- ETL
- SPARQL
- Linked Data

The Basic Approach (1)

Method proposed by Tim Berners-Lee (1998)

Generic, applicable to every database

Automatic

“Table-to-class, column-to-predicate” method

A URI generation scheme also needed

- Should be reversible (i.e. recognize database element from URI)

The Basic Approach (2)

Rules:

(a) Every relation *R* maps to an RDFS class *C*(*R*)

(b) Every tuple of a relation *R* maps to an RDF node of type *C*(*R*)

(c) Every attribute *att* of a relation maps to an RDF property *P*(*att*)

(d) For every tuple *R*[*t*], the value of an attribute *att* maps to a value of the property *P*(*att*) for the node corresponding to the tuple *R*[*t*]

The Basic Approach (3)

Typical URI generation scheme

Database Element	URI Template	Example
Database	*{base_URI}/{db}*	http://www.example.org/company_db
Relation	*{base_URI}/{db}/{rel}*	http://www.example.org/company_db/emp
Attribute	*{base_URI}/{db}/{rel}#{attr}*	http://www.example.org/company_db/emp#name
Tuple	*{base_URI}/{db}/{rel}/{pk=pkval}*	http://www.example.org/company_db/emp/id=5

db: database name

rel: relation name

attr: attribute name

pk: name of a primary key

pkval: value of primary key for given tuple

The Basic Approach (4)

Very crude export

Simple generated ontology

- No complex constructs
- Looks like a copy of the relational schema

New URI for every tuple

- Even when there is an existing one for an entity

All database values mapped to literals

- “Flat” RDF graph

Nevertheless, serves as foundation for several approaches

Creation and Population of a Domain Ontology (1)

"Database schema ontologies" are hardly useful for Linked Data publication

Domain-specific ontologies reflect the domain of the database

Expressiveness of generated ontology depends on the amount of domain knowledge extracted from:

- Human user
- Relational instance

a) Approaches using database schema reverse engineering

b) Basic approach + enrichment from human user

More tools follow b)

- User has full control of the mapping

Creation and Population of a Domain Ontology (2)

Automation level

- Depends on the involvement of the human user

Data accessibility

- SPARQL-based access more popular

Mapping language

- Needed to express complex correspondences between database and ontology
- Until R2RML, every tool used its own language
- Mapping lock-in, low interoperability

Creation and Population of a Domain Ontology (3)

Ontology language

- RDFS, since majority of tools follows basic approach

Vocabulary reuse

- Possible when mappings are manually defined
- User should be familiar with SW vocabularies

Creation and Population of a Domain Ontology (4)

Main goal

- Generate lightweight ontologies reusing existing terms
- Increased semantic interoperability
- Focus not on ontology expressiveness

Motivation

- Mass generation of RDF data from existing large quantities of relational data
- Easier integration with other heterogeneous data

D2RQ / D2R Server (1)

One of the most popular tools in the field

Both automatic and user-assisted operation modes

- Automatic mode
  - Automatic mapping generation
  - Basic approach + rules for M:N relationships $\longrightarrow$ RDFS ontology
- Semi-automatic mode
  - User modifies automatic mapping
- Manual mode
  - User builds mapping from scratch

D2RQ / D2R Server (2)

Custom mapping language

- Feature-rich
  - URI generation mechanism
  - Translation schemes for database values etc.

Both ETL and SPARQL-based access

Vocabulary reuse

- Refer to any ontology inside the mapping file

OpenLink Virtuoso Universal Server

Integration platform (both commercial and open-source versions)

RDF Views feature

- Similar functionality to D2RQ

Both automatic and manual modes

- Automatic mode relies on the basic approach

Virtuoso Meta-Schema language for the mapping definition

- Also very expressive
- One has to learn it in order to customize the mapping (same as in D2RQ)

ETL, SPARQL-based and Linked Data access

Triplify

RDF extraction tool from relational instances

Maps subsets of the database contents (i.e. SQL queries) to URIs of ontology terms

- No need for users to learn a new mapping language

Mappings as configuration files

- Can reuse terms from existing vocabularies (manual editing)

ETL (static) and Linked Data (dynamic) access

Predefined mappings for schemas used by popular Web applications

Supports update logs for RDF resources

- Useful for crawling engines

Ultrawrap

Wraps a database as a SPARQL endpoint

Commercial tool

Supports creation of new domain ontology

- Set of advanced heuristic rules

SPARQL-based access

- SPARQL query refers to terms from new ontology
- Mappings expressed as views defined on the relational schema
- Rewriting to SQL queries referring to above views

Support for manual mappings that reuse terms from existing vocabularies

Oracle DBMS

RDF Views feature (similar to Virtuoso)

Query relational data as RDF

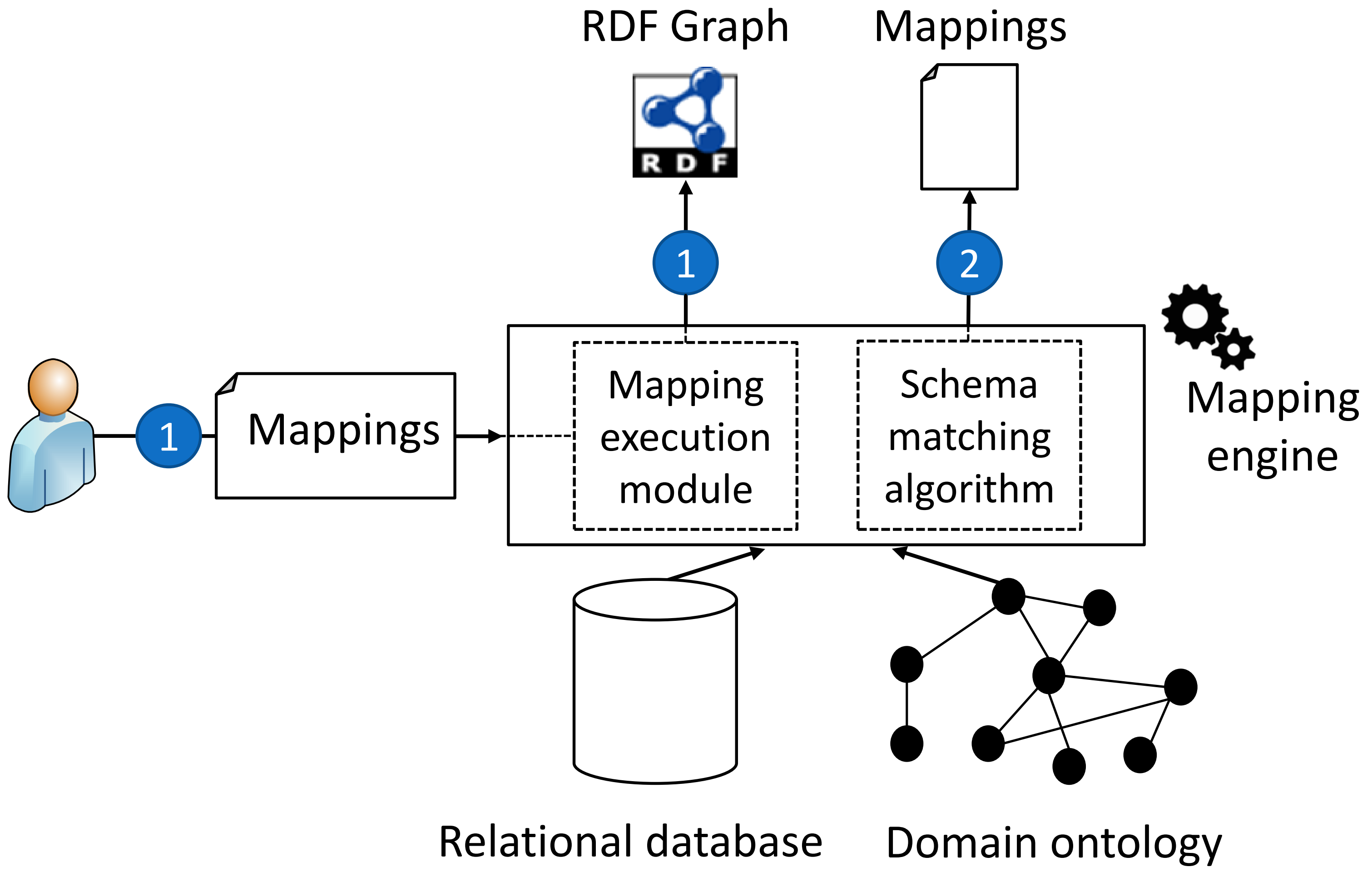
- No replication
- No physical storage for RDF graphs

Both automatic and manual mappings

- Automatic mode follows W3C’s Direct Mapping

Supports combination of virtual and materialized RDF data in the same query

Mapping a Database to an Existing Ontology (1)



Mapping a Database to an Existing Ontology (2)

Existence of ontology is required

- Assumption: Ontology domain same as database domain

Discover mappings between a database and an ontology

- Schema matching algorithms
- Reverse engineering + linguistic similarity measures
- Reuse of such mappings in other applications (e.g. database integration)

Mapping a Database to an Existing Ontology (3)

Apply user-defined mappings to a database

- Mappings refer to one or more existing ontologies
- RDF graph contains instance data from the database
- Tools useful for Linked Data publication

Ontop (1)

Conversion of a relational instance to a SPARQL endpoint

User-defined mappings

Ontology-based data access (OBDA) framework

- Not just SPARQL
- RDFS and OWL 2 QL entailment regimes

Ontop (2)

No need to materialize inferences, calculated at query-time

SPARQL-to-SQL rewriting

- Datalog as intermediate representation language
- Several optimizations simplifying generated SQL queries

Plugin for ontology editor Protégé also available

R_2O / ODEMapster / Morph

Declarative XML-based mapping language

Support for complex mappings
- Conditional mappings
- Definition of URI generation scheme

ODEMapster engine
- R_2O mappings
- Materialized / query-driven access

Morph
- R2RML mappings
- SPARQL-based data access

R2RML Parser

Export of RDF graphs from a relational instance

R2RML mappings

Materialized RDF graph (ETL)

Supports faceted browsing of the generated RDF graph

Incremental dump feature

- Tackles the data synchronization issue
- Graph not generated from scratch
- Only the necessary updates are made to the extracted RDF graph

Outline

Linked Data in Scholarly/Cultural Heritage Domain (1)

Rich experience

Software systems that demonstrate flawless performance

High level of accuracy

Why evolve?

- Data and knowledge description
- New technologies entail new benefits
- Solutions have to remain competitive

Linked Data in Scholarly/Cultural Heritage Domain (2)

Solutions by the LOD paradigm

- Integration
  - Typically materialized using OAI-PMH that does not ease integration with data from other domains
- Expressiveness in describing the information
  - OAI-PMH allows for a tree structure that extends to a depth-level of two
  - RDF allows for a graph-based description
- Query answering
  - Querying graphs using graph patterns allows for much more complex queries

Linked Data in Scholarly/Cultural Heritage Domain (3)

Benefits

- Query expressiveness
- Inherent semantics
- Integration with third party sources

Disadvantages

- Resources investment in creating and maintaining the data

Linked Data in Scholarly/Cultural Heritage Domain (4)

More and more institutions open their data

- Biblioteca Nacional De España
- Deutsche National Bibliothek
- British Library

Linked Data in Scholarly/Cultural Heritage Domain (5)

Is Linked Data the future?

- Content re-use
- Participation of individual collections
- Evolving global Linked Data cloud
- Users can discover new data sources following data-level links
- More complete answers can be delivered as new data sources appear

Ontologies Related to Scholarly Information (1)

Good practice

- Reuse existing vocabularies/ontologies
  - Easier for the outside world to integrate with already existing datasets and services
- Several vocabularies have been proposed

Ontologies Related to Scholarly Information (2)

Title	URL	Namespace	Namespace URL
The Bibliographic Ontology	bibliontology.com	bibo	http://purl.org/ontology/bibo/
Creative Commons Rights Ontology	creativecommons.org	cc	http://creativecommons.org/ns#
CiTo, the Citation Typing Ontology	purl.org/spar/cito	cito	http://purl.org/spar/cito/
Legacy Dublin Core element set	dublincore.org/documents/dces/	dc	http://purl.org/dc/elements/1.1/
DCMI Metadata Terms	dublincore.org/documents/dcmi-terms/	dcterms	http://purl.org/dc/terms/
FaBiO: FRBR-aligned bibliographic ontology	purl.org/spar/fabio	fabio	http://purl.org/spar/fabio/
FRBRcore	purl.org/vocab/frbr/core	frbr	http://purl.org/vocab/frbr/core#
FRBRextended	purl.org/vocab/frbr/extended#	frbre	http://purl.org/vocab/frbr/extended#
IFLA's FRBRer Model	iflastandards.info/ns/fr/frbr/frbrer/	frbrer	http://iflastandards.info/ns/fr/frbr/frbrer/
International Standard Bibliographic Description (ISBD)	iflastandards.info/ns/isbd/elements/	isbd	http://iflastandards.info/ns/isbd/elements/
Lexvo.org Ontology	lexvo.org/ontology	lvont	http://lexvo.org/ontology#
MARC Code List for Relators	id.loc.gov/vocabulary/relators	mrel	http://id.loc.gov/vocabulary/relators/
Open Provenance Model Vocabulary	purl.org/net/opmv/ns	opmv	http://purl.org/net/opmv/ns#
PRISM: Publishing Requirements for Industry Standard Metadata	prismstandard.org	prism	http://prismstandard.org/namespaces/basic/2.0/
Provenance Vocabulary Core Ontology	purl.org/net/provenance/ns	prv	http://purl.org/net/provenance/ns#
RDA Relationships for Works, Expressions, Manifestations, Items	rdvocab.info/RDARelationshipsWEMI	rdarel	http://rdvocab.info/RDARelationshipsWEMI
Schema.org	schema.org	schema	http://schema.org/

Aggregators

International coverage and diverse scope

- European digital heritage gateway Europeana
- DRIVER
- OpenAIRE

Compatibility with aggregators

- Important for repositories
- Common requirement for repositories
- Metadata have to meet specific criteria and adopt specific vocabularies

LOD adoption is the prevailing approach

- Brings an order to the chaos of disparate solutions

Benefits by LOD Adoption

Avoid vendor lock-ins

Allow complex queries to be evaluated on the results

- Utilize the full capacities of SPARQL

Content can be harvested and integrated by third-parties

- Ability to create meta-search repositories
  - Researchers can browse, search and retrieve content from these repositories

Bring existing content into the Semantic Web

- New capabilities are opened

Synchronous Vs. Asynchronous Exports

SPARQL-to-SQL translation in the digital repositories

- Asynchronous approach seems more viable
  - Real-time results may not be as critical
  - RDF updates could take place in a manner similar to search indexes
- The trade-off in data freshness is largely remedied by the improvement in the query answering mechanism
  - Data freshness can be sacrificed in order to obtain much faster results
- Exposing data periodically comes at a low cost
  - Information does not change as frequently as e.g., in sensor data
  - Data is not updated to a significant amount daily
  - Selection queries over the contents are more frequent than the updates

From DSpace to Europeana (1)

DSpace cultural heritage repository

Data model

- Dublin Core
- Europeana Data Model (EDM)

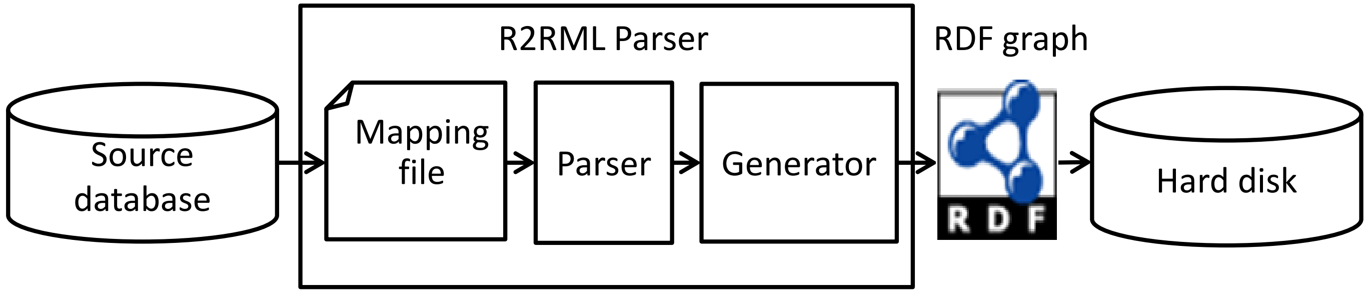
The problem

- How to transform item records as RDF using the EDM model

From DSpace to Europeana (2)

Components

- Source
  - The relational database
- Target
  - An RDF graph
- The R2RML Parser



Information flow

- Parse database contents into result sets
- Generate a Java object
- Instantiates the resulting RDF graph in-memory
- Persist the RDF graph

From DSpace to Europeana (3)

Bibliographic record example

Metadata field	Metadata value
dc.creator	G.C. Zalidis
	A. Mantzavelas
	E. Fitoka
dc.title	Wetland habitat mapping
dc.publisher	Greek Biotope-Wetland Centre
dc.date	1995
dc.coverage.spatial	Thermi
dc.type	Article
dc.rights	http://creativecommons.org/licenses/by/4.0/

From DSpace to Europeana (4)

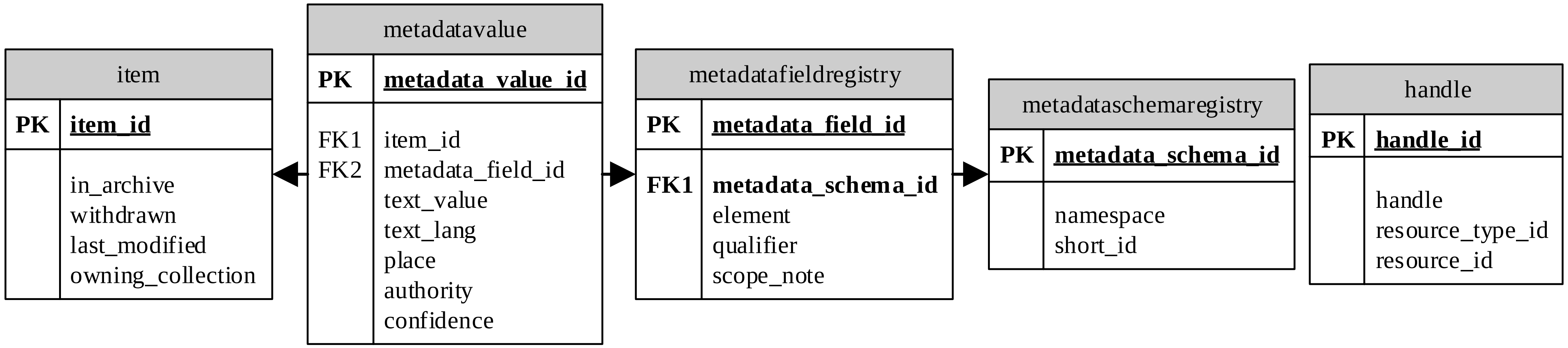
Output description (RDF/XML abbreviated)

```
<edm:ProvidedCHO rdf:about="http://www.example.org/handle/11340/615">
  <dc:creator rdf:resource="http://www.example.org/persons#G.C. Zalidis"/>
  <dc:creator rdf:resource="http://www.example.org/persons#A. Mantzavelas"/>
  <dc:creator rdf:resource="http://www.example.org/persons#E. Fitoka"/>
  <dc:title>
    Wetland habitat mapping
  </dc:title>
  <dc:publisher rdf:resource="http://www.example.org/publishers#Greek Biotope-
Wetland Centre"/>
  <dc:date>1995</dc:date>
  <dcterms:spatial rdf:resource="http://www.example.org/spatial_terms#Thermi"/>
  <dc:type rdf:resource="http://www.example.org/types#Article"/>
  <dc:rights>
    http://creativecommons.org/licenses/by/4.0/
  </dc:rights>
</edm:ProvidedCHO>
```

From DSpace to Europeana (5)

DSpace relational database schema

- Basic infrastructure
- Allows arbitrary schemas and vocabularies



From DSpace to Europeana (6)

Triples Maps definitions in R2RML

Create URIs based on metadata values from Dspace

- Example: dc.coverage.spatial
- Subject (rr:subjectMap template)
  - ' http://www.example.org/handle/{"handle"} '
- Predicate (rr:predicate value)
  - dcterms:spatial
- Object (rr:objectMap template)
  - ' http://www.example.org/spatial_terms#{"text_value"} '

From DSpace to Europeana (7)

R2RML mapping

```
map:dc-coverage-spatial
  rr:logicalTable <#dc-coverage-spatial-view>;
  rr:subjectMap [
    rr:template
'http://www.example.org/handle/{"handle"}';
  ];
  rr:predicateObjectMap [
    rr:predicate dcterms:spatial;
    rr:objectMap [
      rr:template

'http://www.example.org/spatial_terms#{"text_value"}';
      rr:termType rr:IRI
    ];
  ].
```

```
<#dc-coverage-spatial-view>
  rr:sqlQuery """
  SELECT h.handle AS handle, mv.text_value AS
text_value
  FROM handle AS h, item AS i, metadatavalue AS mv,
metadataschemaregistry AS msr, metadatafieldregistry
AS mfr WHERE
  i.in_archive=TRUE AND h.resource_id=i.item_id AND
  h.resource_type_id=2 AND
  msr.metadata_schema_id=mfr.metadata_schema_id AND
  mfr.metadata_field_id=mv.metadata_field_id AND
  mv.text_value is not null AND i.item_id=mv.item_id
AND
msr.namespace='http://dublincore.org/documents/dcmi-
terms/'
  AND mfr.element='coverage' AND
mfr.qualifier='spatial'
  """.
```

From DSpace to Europeana (8)

Technical vs. Bibliographic dimension

Widespread ontologies have to be used where applicable

Linking the data to third party datasets using other datasets' identifiers is also an aspect

Outline

Challenges: Ontology-based Data Updates

SPARQL-based access to the contents of the database is unidirectional

Transform SPARQL Update requests to appropriate SQL statements and execute them on the underlying relational database

An issue similar to the classic database view update problem

Challenges: Mapping Updates

Database schemas and ontologies constantly evolve

- Established mappings should also evolve, not be redefined or rediscovered from scratch

An issue closely related to the previous one

Modifications in either participating model do not incur adaptations to the mapping but cause some necessary changes to the other model

Could prove useful in practice

- Database trigger functions
- The Link Maintenance Protocol (WOD-LMP) from the Silk framework

Challenges: Linking Data

Reusing popular Semantic Web is not sufficient for the generation of 5-star Linked Data

- Database values should not only be translated to RDF literals
- Real-world entities that database values represent should be identified and links between them should be established

Related tools

- RDF extension for Google Refine
- T2LD